

Jake K. Aggarwal
Reneta P. Barneva
Valentin E. Brimkov
Kostadin N. Koroutchev
Elka R. Korutcheva (Eds.)

LNCS 6636

Combinatorial Image Analysis

14th International Workshop, IWCIA 2011
Madrid, Spain, May 2011
Proceedings

 Springer

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

Jake K. Aggarwal Reneta P. Barneva
Valentin E. Brimkov Kostadin N. Koroutchev
Elka R. Korutcheva (Eds.)

Combinatorial Image Analysis

14th International Workshop, IWCIA 2011
Madrid, Spain, May 23-25, 2011
Proceedings

Volume Editors

Jake K. Aggarwal

The University of Texas, Department of Electrical and Computer Engineering
Austin, TX, USA

E-mail: aggarwaljk@mail.utexas.edu

Reneta P. Barneva

State University of New York at Fredonia

Department of Computer and Information Sciences, Fredonia, NY, USA

E-mail: barneva@cs.fredonia.edu

Valentin E. Brimkov

SUNY Buffalo State College, Mathematics Department, Buffalo, NY, USA

E-mail: brimkove@buffalostate.edu

Kostadin N. Koroutchev

Universidad Autónoma de Madrid, Escuela Politécnica Superior

Departamento de Ingeniería Informática, Cantoblanco, Madrid, Spain

E-mail: k.koroutchev@uam.es

Elka R. Korutcheva

Universidad Nacional de Educación a Distancia

Departamento de Física Fundamental, Madrid, Spain

E-mail: elka@fisfun.uned.es

ISSN 0302-9743

e-ISSN 1611-3349

ISBN 978-3-642-21072-3

e-ISBN 978-3-642-21073-0

DOI 10.1007/978-3-642-21073-0

Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2011926787

CR Subject Classification (1998): I.4, I.5, I.3.5, F.2.2, G.2.1, G.1.6

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

© Springer-Verlag Berlin Heidelberg 2011

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

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

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

Printed on acid-free paper

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

Preface

This volume includes the articles presented at the 14th International Workshop on Combinatorial Image Analysis, IWCIA 2011, held in Madrid, Spain, May 23–25, 2011. The 13 previous meetings were held in Paris (France) 1991, Ube (Japan) 1992, Washington D.C. (USA) 1994, Lyon (France) 1995, Hiroshima (Japan) 1997, Madras (India) 1999, Caen (France) 2000, Philadelphia (USA) 2001, Palermo (Italy) 2003, Auckland (New Zealand) 2004, Berlin (Germany) 2006, Buffalo (USA) 2008, and Playa del Carmen (Mexico) 2009.

Over the last 20 years, the IWCIA series has been promoting research on combinatorial approaches to image analysis and processing. In many cases, the proposed solutions have appeared to be more efficient and accurate than those based on continuous models and numeric computation.

Following the call for papers, IWCIA 2011 received 60 submissions. After a rigorous review process, 25 of them were accepted for oral presentation and 13 for poster presentation and for inclusion in this volume. The IWCIA 2011 Program Committee consisted of 81 renowned experts from 26 different countries, and submissions came from about 20 different countries from Asia, Europe, North and South America. The submission and review process was carried out with the help of the *OpenConf* conference system. Review assignments were largely done matching paper topics to topics of expertise indicated by the reviewers. The reviewing process was quite rigorous: each paper received at least three double-blind reviews by members of the Program Committee. The most important selection criterion for acceptance or rejection of a paper was the overall score received. Other criteria included: relevance to the workshop topics, correctness, originality, mathematical depth, clarity, and presentation quality. We believe that as a result, only high-quality papers were accepted for presentation at IWCIA 2011 and for publication in the present volume.

The scientific program of the workshop included oral and poster presentations of contributed papers, as well as keynote talks by four distinguished researchers: Jake Aggarwal, Valentin Brimkov, Concettina Guerra, and Sebastián Cerdán. In addition to the main theoretical track of IWCIA 2011, for the third time a Special Track on Applications was organized. It provided researchers with the opportunity to present their latest achievements and discuss various applications.

The present volume starts with a section containing extended abstracts of the four keynote talks. The contributed papers are grouped into six sections. The first two sections include 16 papers related to digital geometry. The first one is devoted to more general issues, such as digital topology and combinatorics in digital spaces, while the second one is focused on curves and surfaces. The third section contains eight papers on grammars and models for image analysis and related tilings and patterns. The fourth section includes five papers on discrete

tomography. The next one contains five papers on image segmentation, reconstruction, compression, and fuzzy and stochastic image analysis. The last section includes five papers addressing applications in medical imaging and biometrics. We hope that many of these papers are of interest to a broader audience, including researchers working in areas such as computer vision and computer graphics.

We would like to thank everyone who contributed to the success of IWCIAC 2011. First of all, the Chairs are indebted to IWCIAC's Steering Committee for endorsing the candidacy of Spain and Madrid for the 14th edition of the Workshop, as well as to the keynote speakers Jake K. Aggarwal, Valentin E. Brimkov, Concettina Guerra, and Sebastián Cerdán for their remarkable talks and overall contribution to the workshop program.

Our most sincere thanks go to the IWCIAC 2011 Program Committee whose cooperation in carrying out high-quality reviews was essential in establishing a very strong workshop program. We wish to thank everybody who submitted their work to IWCIAC 2011. Thanks to their contributions, we succeeded in having a technical program of high scientific quality. We are indebted to all participants and especially to the contributors of this volume. We hope that the attendees benefited from the scientific program and got inspired with new ideas. We also believe they enjoyed the social program and the excellent conditions provided by the local organizers from the Universidad Autonoma de Madrid. Finally, we express our gratitude to the Springer editorial team, in particular to Alfred Hofmann and Anna Kramer, for their efficient and kind cooperation in the timely production of this book. This book is published with financial support from the Grant TIN2010-11021-E of MICINN, Spain.

May 2011

Jake K. Aggarwal
Reneta P. Barneva
Valentin E. Brimkov
Kostadin Koroutchev
Elka Korutcheva

Organization

IWCIA 2011 was held in Madrid, Spain, May 23–25, 2011.

Honorary Chairs

Jose Dorronsoro	Escuela Politécnica Superior, Vice-Rector of Universidad Autónoma de Madrid, Spain
Estrella Pulido	Dean of the Escuela Politécnica Superior, Universidad Autónoma de Madrid, Spain

General Chair

Kostadin Koroutchev	Universidad Autónoma de Madrid, Spain
---------------------	---------------------------------------

Program and Publication Chair

Reneta P. Barneva	SUNY Fredonia, USA
-------------------	--------------------

Steering Committee

Valentin E. Brimkov	SUNY Buffalo State College, USA
Gabor Herman	CUNY Graduate Center, USA
Kostadin Koroutchev	Universidad Autonoma de Madrid, Spain
Petra Wiederhold	CINVESTAV-IPN, Mexico

Invited Speakers

Jake K. Aggarwal	Univeristy of Texas at Austin, USA
Valentin E. Brimkov	SUNY Buffalo State College, USA
Sebastián Cerdán	Instituto de Investigaciones Biomedicas Alberto Sols, Madrid, Spain
Concettina Guerra	Università di Padova, Italy and Georgia Tech, USA

Program Committee

Til Aach	RWTH Aachen University, Germany
Lyuba Alboul	Sheffield Hallam University, UK
Eric Andres	University of Poitiers, France
Arrate Muñoz	Universidad de Navarra, Spain

VIII Organization

Akira Asano	Hiroshima University, Japan
Jacky Baltès	University of Manitoba, Canada
George Bebis	University of Nevada at Reno, USA
Bedrich Benes	Purdue University, USA
Gilles Bertrand	ESIEE, France
Bhargab B. Bhattacharya	Indian Statistical Institute, India
Peter Brass	City College, City University of New York, USA
Alfred M. Bruckstein	Technion, I.I.T, Israel
Jean-Marc Chassery	University of Grenoble, France
Li Chen	University of the District of Columbia, USA
Marco Cristani	University of Verona, Italy
Guillaume Damiand	LIRIS-CNRS, Université Lyon, France
Leila De Floriani	University of Genova, Italy and University of Maryland, USA
Isabelle Debled-Rennesson	Nancy University, LORIA, France
Eduardo Destefanis	Universidad Tecnológica Nacional-Córdoba, Argentina
Chiou-Shann Fuh	National Taiwan University, Taiwan
Damien Jamet	University of Nancy, France
Jürgen Gall	ETH Zürich Computer Vision Laboratory, Switzerland
Edgar Garduño	IIMAS-UNAM, Mexico
Jordi González i Sabaté	UAB, Spain
Concettina Guerra	Università di Padova, Italy
Edwin Hancock	University of York, UK
Atsushi Imiya	IMIT, Chiba University, Japan
Ramakrishna Kakarala	NTU, Singapore
Andreas Koschan	University of Tennessee, USA
Walter G. Kropatsch	Vienna University of Technology, Austria
Norbert Krüger	Aalborg University Copenhagen, Denmark
Longin Jan Latecki	Temple University, USA
Jerome Liang	SUNY Stony Brook, USA
Pascal Lienhardt	University of Poitiers, France
Shih-Schon Lin	University of Pennsylvania, USA
Joakim Lindblad	Swedish University of Agricultural Sciences, Sweden
Hongbing Lu	Fourth Military Medical University, China
Avner Magen	University of Toronto, Canada
Rémy Malgouyres	Université d'Auvergne, France
Ramon Mas Sansó	Universitat de les Illes Balears, Spain
Erik Melin	Uppsala University, Sweden
Christian Mercat	Université Montpellier, France

Vittorio Murino	University of Verona, Italy
Benedek Nagy	University of Debrecen, Hungary
Akira Nakamura	Hiroshima University, Japan
Renato M. Natal Jorge	University of Porto, Portugal
Gregory M. Nielson	Arizona State University, USA
Janos Pach	City College and Courant Institute, USA
Kálmán Palágyi	University of Szeged, Hungary
Petra Perner	Institute of Computer Vision and Applied Computer Sciences, Germany
Hemerson Pistori	Dom Bosco Catholic University, Brazil
Ioannis Pitas	University of Thessaloniki, Greece
Konrad Polthier	Freie Universität Berlin, Germany
Hong Qin	SUNY Stony Brook, USA
Paolo Remagnino	Kingston University, UK
Ralf Reulke	Humboldt University, Germany
Gerhard Ritter	University of Florida, USA
Mariano Rivera	CIMAT, Mexico
Xavier Roca Marvà	UAB, Spain
Bodo Rosenhahn	MPI Informatik, Germany
Arun Ross	West Virginia University, USA
Angel Sappa	Computer Vision Center, Spain
Henrik Schulz	Forschungszentrum Dresden, Germany
Rani Siromoney	Madras Christian College, India
Isabelle Sivignon	LIRIS-CNRS, University of Lyon, France
Wladyslaw Skarbek	Warsaw University of Technology, Poland
Ali Shokoufandeh	Drexel University, USA
Alberto Soria	CINVESTAV, Mexico
K.G. Subramanian	Universiti Sains Malaysia, Malaysia
Akihiro Sugimoto	National Institute of Informatics, Japan
Mohamed Tajine	University Louis Pasteur, Strasbourg, France
Joao Manuel R.S. Tavares	University of Porto, Portugal
Antonio Turiel	ICM, CSIC, Spain
Peter Veelaert	Ghent University, Belgium
Petra Wiederhold	CINVESTAV-IPN, Mexico
Young Woon Woo	Dong-Eui University Busan, Korea
Jinhui Xu	SUNY University at Buffalo, USA
Yasushi Yagi	Osaka University, Japan
Jason You	Cubic Imaging LLC, USA
Richard Zanibbi	Rochester Institute of Technology, USA
Larry Zeng	University of Utah, USA

Organizing Committee

Elka Korutcheva, Chair	UNED, Spain
Silvia Acuña	UAM, Spain
Alejandro Chinae	UNED, Spain
Ana Fernández	UNED, Spain
Ana González	UAM, Spain
Jaime Moreno	UAM, Spain
Pablo Varona	UAM, Spain

Sponsoring Institutions

Universidad Autónoma de Madrid
Escuela Politécnica Superior

Table of Contents

Invited Papers

Recognition of Human Activities	1
<i>Jake K. Aggarwal</i>	
Complexity and Approximability Issues in Combinatorial Image Analysis	5
<i>Valentin E. Brimkov</i>	
Intelligent Image Analysis of Diffusion Weighted Data Sets: A New Tool for Functional Imaging	9
<i>Blanca Lizarbe, Ania Benitez, Luis Lago, Manuel Sanchez-Montañes, Pilar López-Larrubia, and Sebastián Cerdán</i>	
Computational Methods for the Prediction of Protein-Protein Interactions	13
<i>Concettina Guerra and Marco Mina</i>	

Digital Geometry and Topology, Combinatorics in Digital Spaces

A Family of Topology–Preserving 3D Parallel 6–Subiteration Thinning Algorithms	17
<i>Gábor Németh, Péter Kardos, and Kálmán Palágyi</i>	
On Topology Preservation for Hexagonal Parallel Thinning Algorithms	31
<i>Péter Kardos and Kálmán Palágyi</i>	
Accurate Curvature Estimation along Digital Contours with Maximal Digital Circular Arcs	43
<i>Tristan Roussillon and Jacques-Olivier Lachaud</i>	
Combining Topological Maps, Multi-Label Simple Points, and Minimum-Length Polygons for Efficient Digital Partition Model	56
<i>Guillaume Damiant, Alexandre Dupas, and Jacques-Olivier Lachaud</i>	
Construction of 3D Orthogonal Cover of a Digital Object	70
<i>Nilanjana Karmakar, Arindam Biswas, Partha Bhowmick, and Bhargab B. Bhattacharya</i>	
Skeleton Path Based Approach for Nonrigid 3D Shape Analysis and Retrieval	84
<i>Chunyuan Li and A. Ben Hamza</i>	

The Number of Khalimsky-Continuous Functions between Two Points 96
Shiva Samieinia

Cup Products on Polyhedral Approximations of 3D Digital Images 107
Rocio Gonzalez-Diaz, Javier Lamar, and Ronald Umble

Digital Geometry of Curves and Surfaces

A Jordan Curve Theorem in the Digital Plane 120
Josef Slapal

Maximal Planes and Multiscale Tangential Cover of 3D Digital Objects 132
Emilie Charrier and Jacques-Olivier Lachaud

Recognition of Digital Hyperplanes and Level Layers with Forbidden Points 144
Laurent Provot and Yan Gerard

A Simple and Flexible Mesh Parameterization Method 157
Colin Cartade, Rémy Malgouyres, Christian Mercat, and Chafik Samir

Ellipse Constraints for Improved Wide-Baseline Feature Matching and Reconstruction 168
Dominik Rueß and Ralf Reulke

Reconstruction of Concurrent Lines from Leaning Points 182
Peter Veelaert and Michaël Heyvaert

Isoperimetrically Optimal Polygons in the Triangular Grid 194
Benedek Nagy and Krisztina Barczi

Dynamic Minimum Length Polygon 208
Jacques-Olivier Lachaud and Xavier Provençal

Grammars and Models for Image Analysis.

Tilings and Patterns

On Some Classes of 2D Languages and Their Relations 222
Marcello M. Bersani, Achille Frigeri, and Alessandra Cherubini

Petri Net Generating Hexagonal Arrays 235
D. Lalitha, K. Rangarajan, and D.G. Thomas

Binary Images, M -Vectors, and Ambiguity 248
K.G. Subramanian, Kalpana Mahalingam, Rosni Abdullah, and Atulya K. Nagar

Shuffle on Trajectories over Finite Array Languages	261
<i>H. Geetha, D.G. Thomas, T. Kalyani, and A.S. Prasanna Venkatesan</i>	

Planar Configurations Induced by Exact Polyominoes	275
<i>Daniela Battaglino, Andrea Frosini, and Simone Rinaldi</i>	

Discrete Tomography

Convex-Set Perimeter Estimation from Its Two Projections	284
<i>Étienne Baudrier, Mohamed Tajine, and Alain Daurat</i>	

Solving the Two Color Problem: An Heuristic Algorithm	298
<i>Elena Barcucci, Stefano Brocchi, and Andrea Frosini</i>	

Approximating Bicolored Images from Discrete Projections	311
<i>Fethi Jarray and Ghassen Tlig</i>	

Discrete Q-Convex Sets Reconstruction from Discrete Point X-Rays ...	321
<i>Fatma Abdmouleh, Alain Daurat, and Mohamed Tajine</i>	

Discrete Tomography Reconstruction Based on the Multi-well Potential	335
<i>Tibor Lukić</i>	

Image Segmentation, Representation, Reconstruction, and Compression. Fuzzy and Stochastic Image Analysis

An Optimized Algorithm for the Evaluation of Local Singularity Exponents in Digital Signals	346
<i>Oriol Pont, Antonio Turiel, and Hussein Yahia</i>	

Community Detection for Hierarchical Image Segmentation	358
<i>Arnaud Browet, P.-A. Absil, and Paul Van Dooren</i>	

BCIF: Another Algorithm for Lossless True Color Image Compression	372
<i>Stefano Brocchi and Elena Barcucci</i>	

Distance Measures between Digital Fuzzy Objects and Their Applicability in Image Processing	385
<i>Vladimir Ćurić, Joakim Lindblad, and Nataša Sladoje</i>	

Unsupervised Polygonal Reconstruction of Noisy Contours by a Discrete Irregular Approach	398
<i>Antoine Vacavant, Tristan Roussillon, and Bertrand Kerautret</i>	

Applications to Medical Imaging and Biometrics

Boar Spermatozoa Classification Using Longitudinal and Transversal Profiles (LTP) Descriptor in Digital Images	410
<i>Enrique Alegre, Oscar García-Olalla, Víctor González-Castro, and Swapna Joshi</i>	
Topology-Preserving Registration: A Solution via Graph Cuts	420
<i>Lucilio Cordero-Grande, Gonzalo Vegas-Sánchez-Ferrero, Pablo Casaseca-de-la-Higuera, and Carlos Alberola-López</i>	
Support Vector Machine Approach to Cardiac SPECT Diagnosis	432
<i>Marcin Ciecholewski</i>	
An Entropy-Based Technique for Nonrigid Medical Image Alignment . . .	444
<i>Mohammed Khader and A. Ben Hamza</i>	
Precipitates Segmentation from Scanning Electron Microscope Images through Machine Learning Techniques	456
<i>João P. Papa, Clayton R. Pereira, Victor H.C. de Albuquerque, Cleiton C. Silva, Alexandre X. Falcão, and João Manuel R.S. Tavares</i>	
Nonlinear Dynamical Analysis of Magnetic Resonance Spectroscopy Data	469
<i>Alejandro Chinea</i>	
A Shared Parameter Model for Gesture and Sub-gesture Analysis	483
<i>Manavender R. Malgireddy, Ifeoma Nwogu, Subarna Ghosh, and Venu Govindaraju</i>	
Author Index	495

Recognition of Human Activities

Jake K. Aggarwal

Computer and Vision Research Center
The University of Texas at Austin
Austin, Texas 78712, USA
aggarwaljk@mail.utexas.edu

Computer Vision is the estimation of the three dimensional shape and other properties of objects based on their two dimensional (projection) images through the use of computers and cameras. It had its beginning in the early 1960s. At the time, it was thought to be an easy problem with a solution probably possible over a summer. However, the basic problem has proved to be far more difficult. Over the span of the last 50 years, computer vision has matured from a research topic in the early 1960s to a mature field of research and application. Today, computer vision, image processing, and pattern recognition are addressing many societal and technological problems.

Early interest centered on estimating the shape of inanimate objects like blocks and simple objects such as hand tools. As research progressed in the 1970s, researchers became interested in the motion of objects and in the 1980s started estimating the motion of people. Analysis and understanding of activities followed soon thereafter. The recent desire to monitor people and their activities has led to an added interest in human activity recognition. Security and surveillance applications vary from monitoring persons in public places like a subway station, an airport, a bus station and a parking lot to observing persons in a wide area from a camera mounted on an unmanned aerial vehicle (UAV). Recognition of a person based on static images as well as from video has many applications in surveillance. Monitoring the elderly in a ‘smart’ home equipped with multiple cameras and other sensors is a different type of application. Analysis and understanding of sports video is still another type of activity recognition. Content-based video summarization and retrieval, especially useful to video-sharing websites, is another area with ties to human activity recognition. The movie industry is interested in synthesizing a given persons actions and gait based on a model video of a person. This requires careful capture, analysis, and understanding of the movements of a person and/or a collection of persons. The added constraint in many applications is the need for real-time delivery of the results of processing, analysis, and understanding. At times, this is a truly challenging task.

Advances in several technologies have helped the adoption of computer vision technology in applications. In particular, cameras come in all different sizes, resolutions, shapes, and prices. Webcams are relatively cheap and small and produce

good quality images. Video cameras produce high quality images and may be captured in a computer without difficulty. PTZ cameras with remotely controllable pan, tilt, and zoom capability are readily available. One can use infrared cameras (a bit expensive) or cameras with other modalities for specialized applications. On another technology front, memory and computers have become relatively inexpensive and significantly faster. A person can buy an ‘off the shelf’ system that connects to a home computer and can monitor his/her home from a laptop via the internet. So, the technology is providing an additional boost to computer vision applications.

Before one reaches the stage of high level processing like understanding an activity, many low level image processing steps in a long chain of steps must be performed. In general, these steps pose severe challenges in and of themselves. Images with limited resolution and low contrast pose serious low level images processing difficulties. It is not the intention of the author to soft pedal the difficulties associated with low level processing. A cursory look at the images obtained, for example, in a subway station without the benefit of bright lights will convince a person that low level segmentation is a serious problem. In addition, surveillance is a 24/7 problem, including night-time, rain, and fog if the surveillance is occurring outdoors. In general, low-level segmentation determines the success or failure of the overall system.

The duration of an activity varies with the type of activity and activity is normally a continuous chain of events and not a single event. Given that our recognition methodologies are bottom up in the sense that we recognize “micro activities” or “actions” and then build a concatenation of such recognitions to recognize an activity, several researchers have adopted segmenting an activity at different levels. Earlier the paradigms of ‘change, event, verb, episode, and history’; ‘movement, activity, and action’; and ‘agent, motion, and target’ were used to segment different activities. Other paradigms based on a more flexible context free grammar description of activities have been developed. This has the advantage of describing an activity at the level of detail based on the problem under consideration.

At a gross level, a person is represented as a blob; the level of understanding attainable at this level of granularity is limited to gross level description of motion and activity. One may describe simple actions like depart, follow, and meet. One may construct a system that distinguishes the motion of a person, bicycle, and a vehicle based on the blob and recognize certain gross actions between players. For certain applications, this is adequate. In fact, if one is trying to avoid actual recognition of a person to conform to privacy issues, these techniques are particularly useful. At the next level, a person is represented by body parts namely head, torso, arms legs, hands, and feet. A number of methods have been proposed to address the segmentation of a body into various parts. At times, one is interested in determining the major joints of the body or the extremities of the body since they carry a wealth of information about the activity being performed by the entire body. Semantic recognition of activities based on various body parts has produced some very good results that range from the recognition

of simple activities to the recognition of fighting and recursive activities such as continued fighting. The context of the activity plays an important role in recognition of human activity.

A variety of methodologies are being pursued and one may impose a number of taxonomies to gain insight into methods and results. One such taxonomy is based on dividing the methods into two classes: single layer or hierarchical approaches. In the single layer approach, it may be divided into two cases: space-time or sequential. In the case of hierarchical approaches, recognizing higher levels of activity is based on simpler sub-events related to the activity. The common parts are re-used again in constructing the description of the overall activity. A general purpose recognition system (framework) that can provide a semantic description of diverse human activities is far in the future. Most researchers have focused on special purpose systems addressing particular problems and considered single person activities, two person interactions, and/or crowd activities.

The moving light display experiment of Johansson [5] certainly inspired neuroscience and computer vision based studies of human motion. It motivated Webb [7] to study human motion. A review by the author [1] and other reviews by Cedras and Shah [3], Garvila [4], and Turga et al. [6] and the yet unpublished paper [2] provide an overview of the state of art of recognition of human activities in computer vision. Several problems of computer vision/human activity recognition have proved to be difficult to solve. We have certainly made headway but a solution ala R2D2 is still far in the future. It is fair to assume that some of these problems will be solved in the near future.

Contemporary researchers are addressing a variety of problems involving activity recognition, for example, recognition of human activities from a moving platform. It poses some difficult problems whether it is UAV based camera or a car based camera. In addition to monitoring people, one is naturally interested in the interaction of persons in a scene, the behavior of a crowd, and the possible interaction of a person with movable objects like a piece of luggage or an unmovable object like a fence or a wall. So, working on recognition of human activities has some challenging and interesting ongoing research. Hopefully, this brief abstract provides an introduction to the talk that will provide an exciting array of past accomplishments and future challenges.

References

1. Aggarwal, J.K., Cai, Q.: Human motion analysis: A review. *Computer Vision and Image Understanding* 73(3), 428–440 (1999)
2. Aggarwal, J.K., Ryoo, M.S.: Human activity recognition: A review. *ACM Computing Surveys, CSUR* (to appear, 2011)
3. Cedras, C., Shah, M.: A motion-based recognition: A survey. *Image and Vision Computing* 13(2), 129–155 (1995)
4. Gavrila, D.M.: The visual analysis of human movement: A survey. *Computer Vision and Image Understanding* 73(1), 82–98 (1999)
5. Johansson, G.: Visual perception of biological motion and model for analysis. *Perception Psychophysics* 14(2), 201–211 (1973)

6. Turga, P., Chellappa, R., Subrahmanian, V.S., Udren, O.: Machine recognition of human activities: A survey. *IEEE Transactions on Circuits and Systems for Video Technology* 18(11), 1473–1488 (2008)
7. Webb, J.A., Aggarwal, J.K.: Structure from motion of rigid and jointed objects. *Artificial Intelligence* 19, 107–130 (1982)

Complexity and Approximability Issues in Combinatorial Image Analysis

Valentin E. Brimkov

Mathematics Department, SUNY Buffalo State College,
Buffalo, NY 14222, USA
brimkove@buffalostate.edu

1 Combinatorial Image Analysis and Discrete Geometry

Image analysis is directly applicable to various important and sensitive societal sectors, such as medicine, defense, and security. Often, related research is funded under industrial projects with tight deadlines. Therefore, sometimes certain important theoretical issues remain unaddressed. Such issues have been discussed in length in a recent article [2]. The latter suggested a number of strategic objectives for theoretical research in combinatorial image analysis. Most of these relate to the need to make the discipline better integrated within a number of well-established subjects of theoretical computer science and discrete applied mathematics, such as theory of algorithms and problem complexity, combinatorial optimization and polyhedral combinatorics, integer and linear programming, and computational geometry. Here we concern more in detail one aspect of the research on combinatorial algorithms for image analysis.

Complexity Results and Approximation Algorithms

Theoretical research in combinatorial image analysis is heavily based on structural results and algorithms of discrete geometry. The problems in discrete geometry are combinatorial in nature, which makes them interesting from a point of view of combinatorial optimization and complexity theory.

The theory of algorithms and problem complexity is a central and most important part of theoretical computer science. Complexity issues, therefore, are among the most essential elements distinguishing discrete geometry as an advanced theoretical field.

Many computational problems of discrete geometry have been proved to be NP-hard, which suggests to look for efficient approximation algorithms with guaranteed complexity and accuracy (i.e., together with upper/lower bounds on the algorithms' performance). The performance of a heuristic may need to be analyzed for special probability distributions on inputs. Another possibility is to look for polynomially solvable subclasses of a computationally hard problem. It is also important to investigate the practical versus theoretical performance of an approximation algorithm. As we will illustrate next, poor theoretical performance of an algorithm is not always matched by poor performance in practice.

2 Theoretical vs. Practical Performance: An Example

2.1 Guarding a Set of Segments, Set Cover, and Vertex Cover

Consider any real structure that can be modeled by a set of straight line segments. This can be a network of streets in a city, tunnels in a mine or corridors in a building, pipes in a factory, etc. We want to find a minimal (or close to the minimal) number of locations where to place “guards” (either humans or machines), in a way that any point of the network can be “seen” by at least one guard. Alternatively, we can view this problem as finding a minimal number of devices to place so that a user has access (or a connection) to at least one device from any location. In terms of our mathematical model, we look for a minimum (or close to the minimum) number of points on the given segments, so that every segment contains at least one of them. We call this problem *Guarding a Set of Segments* (GSS).

GSS belongs to the class of the *art gallery problems*. A great variety of such problems have been studied for at least four decades. Related studies have started much earlier by introducing the concepts of starshapedness and visibility (see [6,11]). The reader is referred to the monograph of Joseph O’Rourke [12] and the more recent one of Jorge Urrutia [14]. See also [3,10] and the bibliography therein for a couple of examples of art-gallery problems defined on sets of segments, and [8,1] for possible applications of related studies to efficient wireless communication.

GSS is germane to the set cover (SC), vertex cover (VC) and edge cover (EC) problems¹ (see for details [13]). These are fundamental combinatorial problems that play an important role in complexity theory. It should be noted that we can find applications of GSS anywhere that we find applications of VC where a planar embedding of the graph is relevant or the vertices of the graph represent objects with geometric locations.

GSS can be formulated as a special case of the set cover problem. Under certain restrictions, it can also be formulated as a vertex cover or edge cover problem.

2.2 GSS: Complexity, Polynomial Classes, Approximate Algorithms and Their Theoretical and Experimental Performance

It is well-known that both SC and VC are NP-complete [7,9]. GSS is a special case of the set covering problem where the family of subsets given can be taken as a set of intersections of the given straight line segments. Requiring that the given subsets can be interpreted geometrically this way is a major restriction on the input, yet we have shown that the problem is still strongly NP-complete [4].

¹ Given a universe set U and an arbitrary family of subsets $F \subseteq \mathcal{P}(U)$, the optimization *set cover* problem looks for a minimum cover $C \subseteq F$ so that $\bigcup C = U$.

Given a graph G , a *vertex cover* of G is a set C of vertices of G , such that every edge of G is incident to at least one vertex of C . The optimization vertex cover problem has as an input a graph $G = (V, E)$, and one looks for a vertex cover with a minimal number of vertices.

We have determined certain subclasses of GSS for which the problem admits a polynomial time solution (for example, this is the case for sets of segments that feature a tree structure).

In light of the NP-completeness of the general formulation, we studied the accuracy of three polynomial-time approximation algorithms which return segment coverings. We show that for each of these, theoretically the ratio of the approximate to the optimal solution can increase without bound with the increase of the number of segments. Several other theoretical results concerning GSS approximability have been obtained as well.

In order to study the algorithms performance experimentally, we have used many different pieces of software. The programs can be divided into three components:

- 1) GSS Generators
- 2) GSS Solvers
- 3) Data Analyzers and Visualizers

Surprisingly, our extensive experiments demonstrated that on randomly generated instances the approximate solutions are *always* very close to the optimal ones, and often are, in fact, optimal. See [5] for more details.

2.3 Still Open: What Is GSS Approximability?

An important question remains still open: what is GSS approximability? It is well-known that a minimum set cover can be found in polynomial time within an $O(\log n)$ factor, which is the best possible by order (unless the problems in NP admit quasi-polynomial time solutions). A minimum vertex cover can efficiently be computed within a constant factor. GSS is a special case of set cover and generalization of vertex cover, so its approximability is to be “sandwiched” between the set cover and vertex cover approximabilities. Which of these does the approximation of GSS emulate?

References

1. Aichholzer, O., Fabila-Monroy, R., Flores-Peñaloza, D., Hackl, T., Huemer, C., Urrutia, J., Vogtenhuber, B.: Modem illumination of monotone polygons. In: Proc. 25th European Workshop on Computational Geometry EuroCG 2009, Brussels, Belgium, pp. 167–170 (2009)
2. Asano, T., Brimkov, V.E., Barneva, R.P.: Some theoretical challenges in digital geometry: A perspective. *Discrete Applied Mathematics* 157(16), 3362–3371 (2009)
3. Bose, P., Kirkpatrick, D., Li, Z.: Worst-case-optimal algorithm for guarding planar graphs and polyhedral surfaces. *Computational geometry: Theory and applications* 26(3), 209–219 (2003)
4. Brimkov, V.E., Leach, A., Mastroianni, M., Wu, J.: Guarding a set of line segments in the plane. *Theoretical Computer Science* 412(15), 1313–1324 (2011)

5. Brimkov, V.E., Leach, A., Mastroianni, M., Wu, J.: Experimental study on approximation algorithms for guarding sets of line segments. In: Bebis, G., Boyle, R., Parvin, B., Koracin, D., Chung, R., Hammoud, R., Hussain, M., Kar-Han, T., Crawfis, R., Thalmann, D., Kao, D., Avila, L. (eds.) ISVC 2010. LNCS, vol. 6453, pp. 592–601. Springer, Heidelberg (2010)
6. Brunn, H.: Über Kernegebiete. *Matt. Ann.* 73, 436–440 (1913)
7. Garey, M., Johnson, D.: *Computers and Intractability*. W.H. Freeman & Company, San Francisco (1979)
8. Fabila-Monroy, R., Ruis Vargas, A., Urrutia, J.: On modem illumination problems. In: XIII Encuentros de Geometria Computacional, Zaragoza, Spain (2009), http://www.matem.unam.mx/~urrutia/online_papers/Modems.pdf
9. Karp, R.: Reducibility among combinatorial problems. In: Miller, R.E., Thatcher, J.W. (eds.) *Complexity of Computer Computation*, pp. 85–103. Plenum Press, New York (1972)
10. Āaucic, B., Źalik, B.: A new approach for vertex guarding of planar graphs. *J. of Computing and Information Technology CIT* 10(3), 189–194 (2002)
11. Krasnosel'skii, M.A.: Sur un critère pour qu'un domain soit étoilé. *Mat. Sb.* 19, 309–310 (1946)
12. O'Rourke, J.: *Art Gallery Theorems and Algorithms*. Oxford University Press, Oxford (1987)
13. Papadimitriou, C., Steiglitz, K.: *Combinatorial Optimization*. Prentice-Hall, New Jersey (1982)
14. Urrutia, J.: Art gallery and illumination problems. In: Sack, J.-R., Urrutia, J. (eds.) *Handbook of Computational Geometry*, ch. 22. North Holland, Amsterdam (2000)

Intelligent Image Analysis of Diffusion Weighted Data Sets: A New Tool for Functional Imaging

Blanca Lizarbe¹, Ania Benitez^{1,2}, Luis Lago², Manuel Sanchez-Montañes²,
Pilar López-Larrubia¹, and Sebastián Cerdán¹

¹ Instituto Investigaciones Biomédicas "Alberto Sols",
c/ Arturo Duperier 4, Madrid 28028, Spain

² Departamento de Ingeniería Informática, Escuela Politécnica Superior,
Universidad Autónoma de Madrid, Cantoblanco, Madrid 28039, Spain

1 Introduction

Obesity is a pandemic syndrome underlying the most prevalent causes of death and disability in developed countries including atherosclerosis, ischemic episodes and cancer. Obesity results from an imbalance in global energy metabolism, ultimately caused by disturbances in the neuroendocrine control of appetite in the hypothalamus and leading to an uncompensated feeding/fasting balance [3]. On these grounds, the non invasive detection of hypothalamic activation by food under healthy or diseased conditions entails considerable interest for the diagnosis and treatment of obesity and other food intake disorders as anorexia or bulimia.

Recent evidences support that diffusion weighted imaging (DWI) can provide an independent assessment of stimulated brain areas [2]. The sensitivity of functional DWI to detect micro-structural changes at high b values, on one hand, and the possibility of using IVIM (DWI at low b values) to detect flow changes, on the other, makes this methodology a very suitable technique to explore cerebral hypothalamic activation during a feeding-fasting paradigm in mice.

In this report we propose a systematic approach to analyze DWI data sets including the use of high and low b values to emphasize the different contributions of flow and diffusion effects and implemented a variety of different diffusion models to analyze the data. Our results show that hypothalamic activation is associated to a significant increase in the contribution of the slow diffusion phase in a high b -biexponential diffusion model, and a significant directional-dependent increase in the contribution of the fast diffusion phase in the biexponential diffusion model with low b values. Together, our data provide a novel intelligent environment for the detection and treatment of the cerebral component of obesity.

2 Materials and Methods

Animal model: Adult C57 mice ($n=8$), drinking water *ad libitum* were imaged in two consecutive experimental conditions, fed *ad libitum* and fasted (48 h). *MRI studies:* Mice were maintained anesthetized with 1% isoflurane/oxygen through a nose cap during MRI protocols. We used a 7T Bruker Pharmascan scanner equipped with a 90mm gradient coil insert (36 G/cm, max intensity) and a mouse head resonator.

Diffusion weighting was achieved using the Stejskal-Tanner spin-echo sequence (3) with 4 shot EPI-read gradient and 3 different directions defined by the read, phase and slice encoding gradients. Acquisition conditions were: $\delta = 4ms$, $\Delta = 20ms$, $TR = 3000ms$, $TE = 51ms$, $FOV = 38mm$, axial slices (1.5 mm thickness). We obtained 5 “high b ” value acquisitions ($200 < b < 1200 \text{ s/mm}^2$) and 6 “low b ” value acquisitions ($10 < b < 90 \text{ s/mm}^2$) across an imaging plane containing the hypothalamus. Data analysis: The complete data set was analyzed either using a Linear Discriminant Analysis (LDA) methodology or a biexponential diffusion decay $S(b)/S(0) = SDP \cdot \exp(bD_{slow}) + FDP \cdot \exp(-bD_{fast})$, with slow (SDP) and fast (FDP) diffusion phases characterized by slow (D_{slow}) and fast (D_{fast}) diffusion coefficients. Parameter values were obtained by pixel by pixel fitting of the image data set using home-made MATLAB v7a libraries and compared between fed and fasted states.

3 Results and Discussion

Model free approaches. Figure 1 illustrates the results obtained by Linear Discriminant Analysis as applied to a complete set of diffusion weighted image data sets obtained from representative mouse under the fed or fasted condition. Fisher analysis (Hastie et al. 2001) calculated the vector that optimally separates the pixels from these two conditions. The projection of the values of the fed and fasted conditions of the LDA vector, results in a histogram depicting frequency of occurrence of the projected value under the fed or fasted states (Figure 1, upper panel). The LDA projection can be interpreted as an “appetite index”, with the maximal values for the fed projections and the minimal values for the fasted projections, allowing for the classification of a given pixel between the two possible classes (fed and fasted).

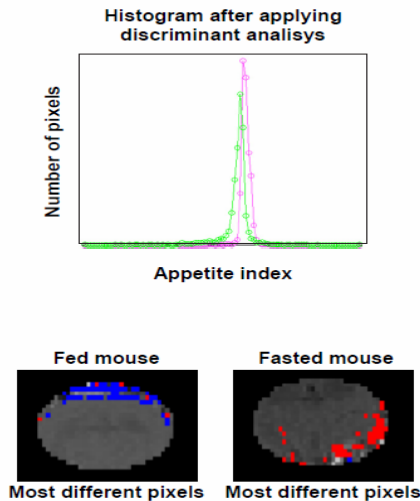


Fig. 1. Histogram after applying LDA to DWI image data sets from the brain of fed and fasted mice (green-fed, purple-fasted, upper panels). Anatomical location of the most representative pixels of between feeding and fasting conditions (lower panels).

The histogram of appetite index values from an animal may be used to decide if the animal is fed or fasted, since the fed and fasted histograms are different. Using this method, we were able to classify correctly the images of all mice between the fed or fasted classes, resulting in 100% correct classifications.

Biexponential Model Approaches. To investigate the physiological causes of the LDA obtained classification, we implemented a biexponential model, as described in Methods. Figure 2 shows representative parameter maps from the Slow Diffusion Phase (SDP) across the hypothalamus of fed and fasted mice for the High b and Low b weightings, superimposed on the corresponding T2-weighted images. For the High b weighting, SDP values increase very appreciably when animals are fasted (in the three diffusion directions measured). Significant differences between feeding conditions are found in the A-P orientation, with $p < 0.001$, and in the H-F direction, with $p < 0.05$. For the Low b weightings, differences are significant in the H-F direction, where SDP calculated values become lower when animals are fasted. In general, the fasted state is described by an increase in the relative contribution of the slow diffusion phase (SDP) component for high b weightings. In the case of low b weightings, we observed an increase in the complementary fast diffusion phase (FDP) component.

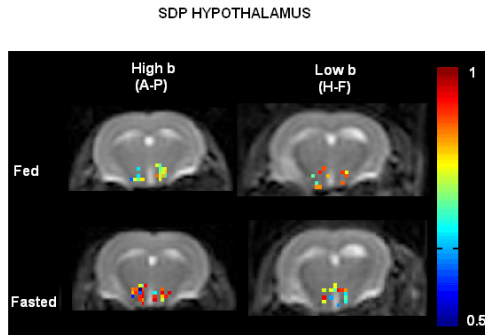


Fig. 2. Representative SDP values in the hypothalamus of fed and fasted mice as determined by the quantitative analysis of DWI using a biexponential model. High b weighting (left) and Low b weighting (right).

4 Conclusion

Taken together, these observations reveal that fasting induces hypothalamic swelling and increased blood flow and these alterations constitute the main physiological differences between the fed and fasted brain as detected in Figure 1. Our results provide in addition, a novel environment for the evaluation of the cerebral component of obesity and may help in the development and implementation of novel anti-obesity therapeutic strategies.

References

1. Hastie, T., Tibshirani, R., Friedman, J.: The elements of statistical learning: Data Mining, Inference, and Prediction. Springer Series in Statistics. Springer, New York (2001)
2. Le Bihan, D.: The 'wet mind': water and functional neuroimaging. *Phys. Med. Biol.* 52, R57–90 (2007)
3. Schwartz, M.W., Woods, S.C., Porte, D., Jr., Seeley, R.J., Baskin, D.G.: Central nervous system control of food intake. *Nature* 404, 661–671 (2000)

Computational Methods for the Prediction of Protein-Protein Interactions

Concettina Guerra^{1,2} and Marco Mina¹

¹ Dept. of Information Engineering, University of Padua, Italy

² College of Computing, Georgia Institute of Technology, USA
{guerra,mina}@dei.unipd.it

Abstract. Broad and extensive knowledge of the biological function of proteins would have great practical impact on the identification of novel drug targets, and on finding the molecular causes of diseases. The experimental in vitro determination of protein function is an expensive and time consuming process. As a consequence, the development of computational techniques to complement and guide the experimental process is a crucial step for biological analysis in the post-genomic era. The prediction of molecular interactions is an important component in functional annotation. Here we shortly describe two approaches to tackle this problem. One approach is structure-based and consists of identifying possible regions of interface on the surface of proteins. A second approach is to transfer a reliable interaction from a species to another species when both species are represented by networks of experimentally determined interactions.

1 Recognition of Proteins Binding Sites

The prediction of interactions is a major task in biology, that has been considered with different types of interacting molecules including protein-protein, protein ligand, protein-RNA and protein-DNA. In particular, the interaction between proteins and ligands is of great interest to the functional annotation of proteins. When a novel protein with unknown function is discovered, bioinformatics tools are used to screen huge datasets of proteins with known function and binding sites, searching for a candidate binding site in the new protein. More specifically, if a surface region of the novel protein is similar to that of the binding site of another protein with known function, the function of the one protein can be inferred and its molecular interaction predicted.

Much work has been done on the analysis of the binding sites of proteins and their identification. This problem is often solved by protein surface matching for which different instances have been considered in the literature:

1. Given binding sites for numerous proteins, the sites are compared and classified [10,13,15].
2. For a given binding site on a first protein, find the surface region of a second protein most similar to the given binding site [8,16].
3. Given two protein surfaces find similar patches on the two surfaces [2].

The techniques employed are numerous ranging from geometric hashing of triangles of points and their associated physico-chemical properties [13], to clique detection on the vertices of the triangulated solvent-accessible surface. Other methods have been developed to identify specific three-dimensional patterns of amino acid side-chains, for instance "catalytic triads" by using for instance graph isomorphism.

We have developed a suite of methodologies and programs for the problem of protein-ligand binding site recognition, based on a representation of the proteins by a collection of spin-images [12]. Experiments conducted on several protein structures from different families and binding to different ligands revealed that in many cases the largest paired regions discovered with high similarity on two protein surfaces actually correspond to the area around the binding site.

Cavity detection is often the first step for functional analysis, since binding sites in proteins usually lie in cavities. Several methods and procedures exist to detect protein cavities, either internal to a molecule or external on a protein surface. The cavity detection algorithms are often based on fitting probe spheres into the spaces between the atoms as in the program SURFNET [6]. We have also developed a method to elucidate protein-protein interactions that identifies cavities on protein surfaces as potential binding sites and matches them [3]. The method has been tested on the dataset that comprised 244 non redundant PDB entries including enzymes (45.9%), nonenzymes (52.9%), and "hypothetical" (1.2%) proteins, according to PDBsum and Uniprot. For the majority of cases, the binding site was correctly found to be on one of the four biggest clefts. We have made our programs available on the web at <http://bcb.dei.unipd.it/MolLoc/>. The programs include a web server for cavity detection and for the comparison of binding sites or cavities.

In [5] a method to quickly identify promising binding sites, either in a protein cavity or on an entire protein surface was proposed based on spherical harmonics. The aim was to efficiently detect putative binding sites without explicitly aligning them, i.e., without actually computing the optimal rotation that best overlaps two binding sites. Instead, with our method we were able to simultaneously evaluate all possible rotations corresponding to a single translation.

2 Protein-Protein Interaction Networks: Analysis and Comparison

Much data has become recently available about protein-protein interaction (PPI) networks in selected organisms by a number of laboratory experiments, including high-throughput techniques, such as yeast-2-hybrid assays, and co-immuno-precipitation. A PPI network is generally modeled by a graph whose nodes correspond to proteins and edges to interactions between proteins. The available PPI data are incomplete and often noisy, thus the graphs are generally rather sparse and their edges not very reliable. The presence of large numbers of false negatives and false positives affects the studies on these networks. Thus,

statistical methods have been designed to systematically validate the actual protein interactions.

The topology of PPI networks is not random and has been characterized in terms of global features, such as degree distributions, average node distance, and clustering coefficients [11]. Other studies have focused on local features of biological networks such as appearance of patterns of local interactions or motifs [12]. Motifs have been defined as both subgraphs that frequently appear in an ensemble of graphs, and subgraphs that are over-represented in a single network. The latter definition assumes a graph model of the given network, thus it assesses the significance of subgraphs frequency with respect to a set of random networks derived by the same model. Graph isomorphism is a required step in most motif discovery algorithms proposed in the literature for counting the occurrences of subgraphs and distinguishing non-isomorphic subgraphs. Sub-graph isomorphism is a NP-complete problem; however, for specific instances of graphs, efficient heuristics can be set up to reduce the search space.

An interesting problem is that of discovering sub-graphs in PPI networks that are highly conserved across species. There is experimental evidence that PPI networks evolve at a modular level. Nodes prefer to attach to well-connected nodes. Furthermore, it has been observed that the interactions among groups of proteins that are temporally close in the course of evolution are likely to be conserved. Consequently, understanding of conserved substructures through alignment of these networks can provide basic insights into a variety of biochemical processes. Network alignment is the problem of finding correspondences between the nodes in two networks based on sequence similarity of the nodes as well as on the similarity of their neighbor's topology. The search for an alignment of two networks is often formulated as a graph optimization problem that incorporates information about the evolution in the definition of the score function to be optimized in the alignment.

Alignment methods can be classified depending on whether they aim at a global [14] or a local alignment [7,9], in other words whether the goal is to determine a measure of similarity between the overall structures of the networks, or to uncover subgraphs of the two graphs with high degree of sequence and topological similarity. Such subgraphs are likely to correspond to functional modules or complexes involved in the same biological process. They usually consist of a set of highly interacting proteins, i.e. dense subgraphs in PPI networks. However, due to the incompleteness of the data, i.e., missing interactions, often they are sparse subgraphs. For this reason, a recently designed approach [4] addresses the problem of identifying conserved complexes by searching for relatively dense groups of nodes with only sparser connections outside.

Comparative analysis is a very powerful tool for the prediction of protein-protein interactions, based on the fact that their corresponding proteins in the alignment interact within another species. Network alignment is also useful for functional prediction since the annotations from the annotated proteins can be transferred to the aligned proteins which are not annotated in another organism.

References

1. Angaran, S., Bock, M.E., Garutti, C., Guerra, C.: MolLoc: a web tool for the local alignment of molecular surfaces. *Nucleic Acids Research* (2009), doi:10.1093/NAR/GKP405
2. Bock, M.E., Garutti, C., Guerra, C.: Discovery of similar regions on protein surfaces. *Journal of Computational Biology* 14(3), 285–299 (2007)
3. Bock, M.E., Garutti, C., Guerra, C.: Cavity detection and Binding site recognition in proteins. *Theoretical Computer Science* (2008), doi:doi:10.1016/j
4. Cannataro, M., Ciriello, G., Guerra, C., Guzzi, P., Mina, M.: AlignNemo: a novel method to align PPI networks (manuscript)
5. Comin, M., Dellaert, F., Guerra, C.: Binding Balls: Fast detection of binding sites using a property of Spherical Fourier Transform. *Journal of Computational Biology* 16 (2009), doi:10.1089/cmb.2009.0045
6. Glaser, F., Morris, R.J., Najmanovich, R.J., Laskowski, A., Thornton, J.M.: A method for localizing ligand binding pockets in protein structures. *Proteins: Struct. Funct. Bioinf.* 62, 479–488 (2006)
7. Kelley, B.P., et al.: Conserved pathways within bacteria and yeast as revealed by global protein network alignment. *Proc. Natl. Acad. Sci. USA* 100, 11394–11399 (2003)
8. Kinoshita, N., Furui, J., Nakamura, H.: Identification of protein functions from a molecular surface database, eF-site. *J. Struct. Funct. Genomics* 2, 9–22 (2001)
9. Koyuturk, M., et al.: Pairwise alignment of protein interaction networks. *J. Comput. Biol.* 13, 182–199 (2006)
10. Morris, R.J., Najmanovich, R.J., Kahraman, A., Thornton, J.M.: Real spherical harmonic expansion coefficients as 3D shape descriptors for protein binding pocket and ligand comparison. *Bioinformatics* 21(10), 2347–2355 (2005)
11. Newman, M.E.J.: The structure and function of complex networks. *SIAM Rev.* 45(2), 167–256 (2003)
12. Shen-Orr, S.S., Milo, R., Mangan, S., et al.: Network motifs in the transcriptional regulation network of *Escherichia coli*. *Nat. Genet.* 31, 64–68 (2002)
13. Shulman-Peleg, A., Nussinov, R., Wolfson, H.J.: Recognition of functional sites in protein structures. *J. Mol. Biol.* 339, 607–633 (2004)
14. Singh, R., Xu, J., Berger, B.: Pairwise global alignment of protein interaction networks by matching neighborhood topology. In: Speed, T., Huang, H. (eds.) *RECOMB 2007*. LNCS (LNBI), vol. 4453, pp. 16–31. Springer, Heidelberg (2007)
15. Sommer, I., Miller, O., Domingues, F., Sander, O., Weickert, J., Lengauer, T.: Moment invariants as shape recognition technique for comparing protein binding sites. *Bioinformatics* 23, 3139–3146 (2007)
16. Yao, H., Kristensen, D.M., Mihalek, I., Sowa, M.E., Shaw, C., Kimme, M., Kaviraki, L., Lichtarge, O.: An accurate, sensitive, and scalable method to identify functional sites in protein structures. *J. Mol. Biol.* 326, 255–261 (2003)

A Family of Topology–Preserving 3D Parallel 6–Subiteration Thinning Algorithms

Gábor Németh, Péter Kardos, and Kálmán Palágyi

Department of Image Processing and Computer Graphics,
University of Szeged, Hungary
{gnemeth,pkardos,palagyi}@inf.u-szeged.hu

Abstract. Thinning is an iterative layer-by-layer erosion until only the skeleton-like shape features of the objects are left. This paper presents a family of new 3D parallel thinning algorithms that are based on our new sufficient conditions for 3D parallel reduction operators to preserve topology. The strategy which is used is called subiteration-based: each iteration step is composed of six parallel reduction operators according to the six main directions in 3D. The major contributions of this paper are: 1) Some new sufficient conditions for topology preserving parallel reductions are introduced. 2) A new 6–subiteration thinning scheme is proposed. Its topological correctness is guaranteed, since its deletion rules are derived from our sufficient conditions for topology preservation. 3) The proposed thinning scheme with different characterizations of endpoints yields various new algorithms for extracting centerlines and medial surfaces from 3D binary pictures.

Keywords: shape representation, skeletonization, thinning, topology preservation.

1 Introduction

Skeleton-like shape features (i.e., centerline, medial surface, and topological kernel) extracted from 3D binary images play an important role in numerous applications of image processing and pattern recognition [19].

Parallel thinning algorithms [4] are capable of extracting skeleton-like shape descriptors in a topology preserving way [6]. Their iteration steps are composed of some parallel reduction operators: some object points having value of “1” in a binary image that satisfy certain topological and geometric constraints are deleted (i.e., changed to “0” ones) simultaneously, and the entire process is repeated until no points are deleted.

An object point is simple if its deletion does not alter the topology of the image [6]. In a phase of a parallel thinning algorithm, a set of simple points is deleted simultaneously that may not preserve the topology. A possible approach to overcome this problem is to use subiteration-based thinning (often referred to as directional or border sequential strategy) [4]: each iteration step is composed of k subiterations ($k \geq 2$), where only border points of certain kind are deleted.

Since there are six major directions in 3D, most of existing parallel 3D directional thinning algorithms use six subiterations [3,14].

Object points having value of “1” in a binary image are endpoints if they provide important geometrical information relative to the shape of the objects to be represented. Surface-thinning algorithms are to extract *medial surfaces* by preserving *surface-endpoints*, curve-thinning algorithms produce *centerlines* by preserving *curve-endpoints*, and *topological kernels* (i.e., minimal structures which are topologically equivalent to the original objects) can be generated if no endpoint characterization is considered during the thinning process [2]. Medial surfaces are usually extracted from general shapes, tubular structures can be represented by their centerlines, and extracting topological kernels are useful in topological description.

The deletion rules of existing parallel thinning algorithms are generally given by matching templates with specific and “built-in” endpoint characterizations [1,3,8,9,10,11,14,15,16,20] with the exceptions of some 3D fully parallel algorithms [17] and some 3D subfield-based thinning algorithms [12,13]. In this paper, we introduce a general scheme for 6-subiteration 3D parallel thinning that is based on our new sufficient conditions for topology preservation. The proposed scheme coupled with different types of endpoints yields various topology preserving thinning algorithms.

The rest of this paper is organized as follows. Section 2 gives the basic notions of 3D digital topology. Then in Section 3 we propose our sufficient conditions for 3D parallel reduction operators to preserve topology. Section 4 presents a family of new 6-subiteration 3D parallel thinning algorithms. Finally, Section 5 gives five variations for the proposed thinning scheme by considering five different characterizations of endpoints.

2 Basic Notions and Results

Let p be a point in the 3D digital space \mathbb{Z}^3 . Let us denote $N_j(p)$ (for $j = 6, 18, 26$) the set of points that are j -adjacent to point p (see Fig. 1a).

The 3D binary $(26, 6)$ digital picture \mathcal{P} is a quadruple $\mathcal{P} = (\mathbb{Z}^3, 26, 6, X)$ [6], where each element of \mathbb{Z}^3 is called a *point* of \mathcal{P} , each point in $X \subseteq \mathbb{Z}^3$ is called a *black point* and it has a value of “1”, each point in $\mathbb{Z}^3 \setminus X$ is called a *white point* and value of “0” is assigned to it. 26-connectivity (i.e., the reflexive and transitive closure of the 26-adjacency relation) is considered for black points forming the objects, and 6-connectivity (i.e., the reflexive and transitive closure of the 6-adjacency) is considered for white points [6] (see Fig. 1a). Maximal 26-connected components of black points are called *objects*.

A black point is called a *border point* in a $(26, 6)$ picture if it is 6-adjacent to at least one white point. A border point p is called a **U**-border point if the point marked $\mathbf{U} = u(p)$ in Fig. 1a is a white point. We can define **D**-, **N**-, **E**-, **S**-, and **W**-border points in the same way. A black point is called an *interior point* if it is not a border point. There are three *opposite pairs* **U**-**D**, **N**-**S**, and **E**-**W** in $N_6(p) \setminus \{p\}$.

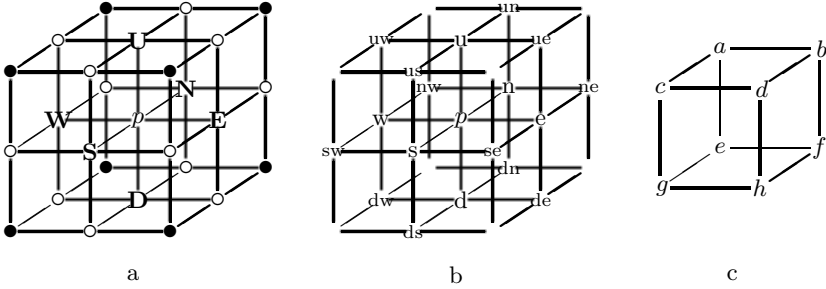


Fig. 1. Frequently used adjacencies in \mathbb{Z}^3 (a). The set $N_6(p)$ contains point p and the six points marked **U**, **D**, **N**, **E**, **S**, and **W**. The set $N_{18}(p)$ contains $N_6(p)$ and the twelve points marked “o”. The set $N_{26}(p)$ contains $N_{18}(p)$ and the eight points marked “•”. Notation for the points in $N_{18}(p)$ (b). The $2 \times 2 \times 2$ cube that contains an object (c)

A *parallel reduction operator* changes a set of black points to white ones (which is referred to as deletion). A 3D parallel reduction operator does *not* preserve topology if any object is split or is completely deleted, any cavity (i.e., maximal 6-connected component of white points) is merged with another cavity, a new cavity is created, or a hole (that donuts have) is eliminated or created.

A black point is called a *simple point* if its deletion does not alter the topology of the image [6]. Note that simplicity of point p in (26, 6) pictures is a local property that can be decided by investigating the set $N_{26}(p)$ [6].

Parallel reduction operators delete a set of black points and not only a single simple point. Ma gave some *sufficient conditions* for 3D parallel reduction operators to preserve topology [7]. Those conditions require some additional concepts to be defined. Let \mathcal{P} be a (26, 6) picture. The set $D = \{d_1, \dots, d_k\}$ of black points is called a *simple set* of \mathcal{P} if D can be arranged in a sequence $\langle d_{i_1}, \dots, d_{i_k} \rangle$ in which d_{i_1} is simple and each d_{i_j} is simple after $\{d_{i_1}, \dots, d_{i_{j-1}}\}$ is deleted from \mathcal{P} , for $j = 2, \dots, k$. (By definition, let the empty set be simple.) A *unit lattice square* is a set of four mutually 18-adjacent points in \mathbb{Z}^3 ; a *unit lattice cube* is set of eight mutually 26-adjacent points in \mathbb{Z}^3 .

Theorem 1. [7] *A 3D parallel reduction operator is topology preserving for (26,6) pictures if all of the following conditions hold:*

1. *Only simple points are deleted.*
2. *If two, three, or four black corners of a unit lattice square are deleted, then these corners form a simple set.*
3. *No object contained in a unit lattice cube is deleted completely.*

3 New Sufficient Conditions for Topology Preserving Parallel Reductions

Theorem 1 provides a general method of verifying that a parallel thinning algorithm preserves topology [5]. In this section, we present some new sufficient

conditions for topology preservation as a basis for designing 3D 6-subiteration parallel thinning algorithms. In order to introduce our new sufficient conditions for topology preserving parallel reductions that delete **U**-border points, we define two special kinds of point sets.

Definition 1. Let $p \in X$ be a black point in picture $(\mathbb{Z}^3, 26, 6, X)$ and let $S(p) \subseteq X \setminus \{p\}$ be a set of black points such that $S(p) \cup \{p\}$ is contained in a unit lattice square. The set $S(p)$ is called a **U**-square-considerable set if for any point $s \in S(p) \cup \{p\}$, $u(s) \notin S(p) \cup \{p\}$.

We can define **D**-, **N**-, **E**-, **S**-, and **W**-square-considerable sets in the same way. Let us state some properties of **U**-square-considerable sets.

Proposition 1. The following 33 sets may be **U**-square-considerable ones (see Fig. [1b](#)):

\emptyset , $\{un\}$, $\{ue\}$, $\{us\}$, $\{uw\}$, $\{nw\}$, $\{n\}$, $\{ne\}$, $\{w\}$, $\{e\}$, $\{sw\}$, $\{s\}$,
 $\{se\}$, $\{dn\}$, $\{de\}$, $\{ds\}$, $\{dw\}$, $\{nw,n\}$, $\{nw,w\}$, $\{n,w\}$, $\{ne,n\}$,
 $\{ne,e\}$, $\{n,e\}$, $\{sw,s\}$, $\{sw,w\}$, $\{s,w\}$, $\{se,s\}$, $\{se,e\}$, $\{s,e\}$,
 $\{nw,n,w\}$, $\{ne,n,e\}$, $\{sw,s,w\}$, $\{se,s,e\}$.

Proposition 2. Any subset of a **U**-square-considerable set is a **U**-square-considerable set as well.

These properties are obvious by careful examination of the points in $N_{18}(p)$ (see Fig. [1b](#)).

Definition 2. Let $C \subseteq X$ be an object of picture $(\mathbb{Z}^3, 26, 6, X)$ that is contained in a unit lattice cube. C is called a **U**-cube-considerable object if all of the following conditions hold:

1. $\#(C) \geq 2$ (where $\#(C)$ denotes the number of elements in C).
2. For any point $c \in C$, $u(c) \notin C$ (i.e., C must contain **U**-border points).
3. C is not contained in a unit lattice square.

We can define **D**-, **N**-, **E**-, **S**-, and **W**-cube-considerable objects in the same way. Let us state the two most important properties of **U**-cube-considerable objects.

Proposition 3. For any **U**-cube-considerable object C , $\#(C) \leq 4$.

It is easy to see that any object contained in a unit lattice cube that contains 5, 6, 7, or 8 points, must contain at least one element that is not a **U**-border point (i.e., it must contain a pair of points p and $u(p)$).

Proposition 4. There are 32 possible **U**-cube-considerable objects.

The possible **U**-cube-considerable objects are listed as follows (see Fig. [1c](#)):

$\{a, h\}$, $\{a, h, b\}$, $\{a, h, b, c\}$, $\{a, h, b, g\}$, $\{a, h, c\}$, $\{a, h, c, f\}$, $\{a, h, f\}$, $\{a, h, f, g\}$,
 $\{a, h, g\}$, $\{b, g\}$, $\{b, g, a\}$, $\{b, g, a, d\}$, $\{b, g, d\}$, $\{b, g, d, e\}$, $\{b, g, e\}$, $\{b, g, e, h\}$,
 $\{b, g, h\}$, $\{c, f\}$, $\{c, f, a\}$, $\{c, f, a, d\}$, $\{c, f, d\}$, $\{c, f, d, e\}$, $\{c, f, e\}$, $\{c, f, e, h\}$,
 $\{c, f, h\}$, $\{d, e\}$, $\{d, e, b\}$, $\{d, e, b, c\}$, $\{d, e, c\}$, $\{d, e, f\}$, $\{d, e, f, g\}$, $\{d, e, g\}$.

The lexicographical order relation “ \prec ” between two distinct points $p = (p_x, p_y, p_z)$ and $q = (q_x, q_y, q_z)$ is defined as follows:

$$p \prec q \iff (p_z < q_z) \vee (p_z = q_z \wedge p_y < q_y) \vee (p_z = q_z \wedge p_y = q_y \wedge p_x < q_x).$$

Definition 3. Let $C \subseteq \mathbb{Z}^3$ be a set of points. Point $p \in C$ is the smallest element of C if for any $q \in C \setminus \{p\}$, $p \prec q$.

We are now ready to state our new sufficient conditions for topology preserving parallel reductions that delete **U**-border points. Note that sufficient conditions for simultaneous deletion of **D**-, **N**-, **E**-, **S**-, and **W**-border points can be given in the same way.

Theorem 2. Let T be a parallel reduction operator. Let p be any black point in any picture $(\mathbb{Z}^3, 26, 6, X)$ such that point p is deleted by T . Operator T is topology preserving for $(26, 6)$ pictures if all of the following conditions hold:

1. Point p is a simple and **U**-border point in picture $(\mathbb{Z}^3, 26, 6, X)$.
2. For any **U**-square-considerable set $S(p)$ that contains simple and **U**-border points in $(\mathbb{Z}^3, 26, 6, X)$, p is a simple point in picture $(\mathbb{Z}^3, 26, 6, X \setminus S(p))$.
3. Point p is not the smallest element of any **U**-cube-considerable object.

Proof. To prove it, we show that the parallel reduction operator T satisfies all conditions of Theorem 1.

1. Operator T may delete simple points by Condition 1 of Theorem 2. Hence Condition 1 of Theorem 1 is satisfied.
2. Since operator T may delete **U**-border points (by Condition 1 of Theorem 2), it is sufficient to deal with the 33 possible **U**-square-considerable sets (see Definition 1, Proposition 1, and Proposition 2). The following points have to be checked:
 - (a) Suppose that $S(p) = \emptyset$ ($\#(S(p)) = 0$). Since Condition 1 of Theorem 2 holds, point p is simple in $(\mathbb{Z}^3, 26, 6, X) = (\mathbb{Z}^3, 26, 6, X \setminus S(p))$. Therefore, Condition 2 of Theorem 1 is satisfied.
 - (b) Let a and b be two corners of a unit lattice square that are deleted by T . If $p = b$ and $S(p) = \emptyset$, then b is a simple point in $(\mathbb{Z}^3, 26, 6, X)$ by case (a). Suppose that $p = a$ and $S(p) = \{b\}$. Since Condition 2 of Theorem 2 holds, point a is simple in $(\mathbb{Z}^3, 26, 6, X \setminus S(p))$. Consequently, $\{a, b\}$ is a simple set. Therefore, Condition 2 of Theorem 1 is satisfied.
 - (c) Let a, b , and c be three corners of a unit lattice square that are deleted by T . In this case b and c are two corners of a unit lattice square and $\{b, c\}$ is a simple set by case (b). Suppose that $p = a$ and $S(p) = \{b, c\}$. Since Condition 2 of Theorem 2 holds, point a is simple in $(\mathbb{Z}^3, 26, 6, X \setminus S(p))$. Consequently, the set $\{a, b, c\}$ is simple. Therefore, Condition 2 of Theorem 1 is satisfied.
 - (d) Let a, b, c , and d be four corners of a unit lattice square that are deleted by T . In this case b, c and d are three corners of a unit lattice square

and $\{b, c, d\}$ is a simple set by case (c). Suppose that $p = a$ and $S(p) = \{b, c, d\}$. Since Condition 2 of Theorem 2 holds, point a is simple in $(\mathbb{Z}^3, 26, 6, X \setminus S(p))$. Consequently, the set $\{a, b, c, d\}$ is simple. Therefore, Condition 2 of Theorem 1 is satisfied.

3. Let us consider object C that is contained in a unit lattice cube. The following points have to be checked:
- (a) Suppose that $\#(C) = 1$, $C = \{a\}$. In this case, a is an isolated point that is not simple. Since Condition 1 of Theorem 2 holds, point a cannot be deleted by T . Therefore, Condition 3 of Theorem 1 is satisfied.
 - (b) Suppose that $\#(C) = 2$, $C = \{a, b\}$. If a and b are two corners of a unit lattice square, then C cannot be deleted completely by Condition 2 of Theorem 2. If C contains a point that is not a \mathbf{U} -border point, then C cannot be deleted completely by Condition 1 of Theorem 2. Otherwise C is a \mathbf{U} -cube-considerable object and its smallest element cannot be deleted by Condition 3 of Theorem 2. Therefore, Condition 3 of Theorem 1 is satisfied.
 - (c) Suppose that $\#(C) = 3$, $C = \{a, b, c\}$. If a , b , and c are three corners of a unit lattice square, then C cannot be deleted completely by Condition 2 of Theorem 2. If C contains a point that is not a \mathbf{U} -border point, then C cannot be deleted completely by Condition 1 of Theorem 2. Otherwise C is a \mathbf{U} -cube-considerable object and its smallest element cannot be deleted by Condition 3 of Theorem 2. Therefore, Condition 3 of Theorem 1 is satisfied.
 - (d) Suppose that $\#(C) = 4$, $C = \{a, b, c, d\}$. If a , b , c , and d are four corners of a unit lattice square, then C cannot be deleted completely by Condition 2 of Theorem 2. If C contains a point that is not a \mathbf{U} -border point, then C cannot be deleted completely by Condition 1 of Theorem 2. Otherwise C is a \mathbf{U} -cube-considerable object and its smallest element cannot be deleted by Condition 3 of Theorem 2. Therefore, Condition 3 of Theorem 1 is satisfied.
 - (e) Suppose that $\#(C) > 4$. In this case, C must contain at least one point that is not a \mathbf{U} -border point by Proposition 3. That point cannot be deleted by Condition 1 of Theorem 2. Therefore, Condition 3 of Theorem 1 is satisfied. \square

4 The New 6-Subiteration Thinning Algorithms

Now we propose a set of new 6-subiteration 3D parallel thinning algorithms. Their deletable points are derived directly from Theorem 2.

Let us consider an arbitrary characterization of endpoints that is called as type \mathcal{E} . The algorithm denoted by **6SI- \mathcal{E}** is our 6-subiteration 3D parallel thinning algorithm that preserves endpoints of type \mathcal{E} (see Algorithm 1).

The usual ordered list of the deletion directions $\langle \mathbf{U}, \mathbf{D}, \mathbf{N}, \mathbf{E}, \mathbf{S}, \mathbf{W} \rangle$ [3, 14] is considered in Algorithm **6SI- \mathcal{E}** . Note that subiteration-based thinning algorithms are not invariant under the order of deletion directions (i.e., choosing different orders may yield various results).

Algorithm 1

```

Input: picture  $(\mathbb{Z}^3, 26, 6, X)$ 
Output: picture  $(\mathbb{Z}^3, 26, 6, Y)$ 
 $Y = X$ 
repeat
  // one iteration step
  for each  $i \in \{\mathbf{U}, \mathbf{D}, \mathbf{N}, \mathbf{E}, \mathbf{S}, \mathbf{W}\}$  do
    // subiteration for deleting some  $i$ -border points
     $D(i) = \{ p \mid p \text{ is an } i\text{-}\mathcal{E}\text{-deletable point in } Y \}$ 
     $Y = Y \setminus D(i)$ 
until  $D(\mathbf{U}) \cup D(\mathbf{D}) \cup D(\mathbf{N}) \cup D(\mathbf{E}) \cup D(\mathbf{S}) \cup D(\mathbf{W}) = \emptyset$ 

```

In the first subiteration of our 6-subiteration thinning algorithms, the set of \mathbf{U} - \mathcal{E} -deletable points are deleted simultaneously, and the set of \mathbf{W} - \mathcal{E} -deletable points are deleted in the last (i.e., the 6th) subiteration. Now we lay down \mathbf{U} - \mathcal{E} -deletable points. We can define \mathbf{D} -, \mathbf{N} -, \mathbf{E} -, \mathbf{S} -, and \mathbf{W} - \mathcal{E} -deletable points in the same way.

Definition 4. A black point p in picture $(\mathbb{Z}^3, 26, 6, X)$ is \mathbf{U} - \mathcal{E} -deletable if all of the following conditions hold:

1. Point p is a simple and \mathbf{U} -border point, but it is not an endpoint of type \mathcal{E} in picture $(\mathbb{Z}^3, 26, 6, X)$.
2. For any \mathbf{U} -square-considerable set $S(p)$ composed of simple points and \mathbf{U} -border points, but not endpoints of type \mathcal{E} in picture $(\mathbb{Z}^3, 26, 6, X)$, point p remains simple in picture $(\mathbb{Z}^3, 26, 6, X \setminus S(p))$.
3. Point p is not the smallest element of any \mathbf{U} -cube-considerable object.

We can state our main theorem.

Theorem 3. Algorithm **6SI- \mathcal{E}** is topology preserving for $(26, 6)$ pictures for arbitrary characterization of endpoints.

Proof. It can readily be seen that Condition i of Definition 4 satisfies Condition i of Theorem 2 ($i = 1, 2, 3$). Consequently, the first subiteration of Algorithm **6SI- \mathcal{E}** is a topology preserving parallel reduction for $(26, 6)$ pictures for arbitrary characterization of endpoints.

Similarly, it can be seen that the five parallel reductions assigned to the remaining five subiterations of Algorithm **6SI- \mathcal{E}** are topology preserving as well. Hence, the entire algorithm composed of topology preserving reductions is topology preserving too.

Note that the proof of Theorem 2 does not consider the applied type of endpoints \mathcal{E} . Hence arbitrary characterizations of endpoints yield topologically correct 6-subiteration thinning algorithms. \square

5 Examples of the New 6-Subiteration Thinning Algorithms

In Section 3, we defined the deletable points of the proposed 6-subiteration thinning algorithm **6SI- \mathcal{E}** that preserves endpoints of type \mathcal{E} . We stated that various characterizations of endpoints yield different algorithms. Here, we define four types of endpoints (**C1**, **C2**, **S1**, and **S2**) that determine four new thinning algorithms (**6SI-C1**, **6SI-C2**, **6SI-S1**, and **6SI-S2**). Furthermore, if no endpoints are preserved, then we get topological kernels. Therefore, no restriction is applied to an “endpoint” of type **TK**, which leads to the algorithm called **6SI-TK**.

Definition 5. A “1” point p in picture $(\mathbb{Z}^3, 26, 6, X)$ is a curve-endpoint of type **C1** if $(N_{26}(p) \setminus \{p\}) \cap X = \{q\}$ (i.e., p is 26-adjacent to exactly one “1” point).

Definition 6. A “1” point p in picture $(\mathbb{Z}^3, 26, 6, X)$ is a curve-endpoint of type **C2** if $(N_{26}(p) \setminus \{p\}) \cap X = \{q\}$ and the number of elements in $(N_{26}(q) \setminus \{q\}) \cap X$ is less than or equal to 2.

Definition 7. A “1” point p in picture $(\mathbb{Z}^3, 26, 6, X)$ is a surface-endpoint of type **S1** if there is no interior point in the set $N_6(p) \cap X$.

Note the characterization of surface-endpoints **S1** are applied in some existing thinning algorithms [11][16].

Definition 8. A “1” point p in picture $(\mathbb{Z}^3, 26, 6, X)$ is a surface-endpoint of type **S2** if the set $N_6(p) \setminus \{p\}$ contains at least one opposite pair of “0” points.

Note that the characterization of surface-endpoints **S2** is introduced in [15].

In experiments algorithm **6SI-TK** and the further algorithms based on the four types of endpoints according to Definitions 5-8 were tested on objects of different shapes. Here we present some illustrative examples below (Figs. 2[8]). Our new algorithms are compared with the existing 6-subiteration curve-thinning algorithm **PK-C** [14] and surface thinning algorithm **GB-S** [3]. Numbers in parentheses mean the count of “1” points.

The tubular test objects in Figs. 2[4] are represented by their centerlines extracted by the three curve-thinning algorithms **6SI-C1**, **6SI-C2**, and **PK-C**.

We can state that algorithm **6SI-C2** produces less skeletal points than algorithm **6SI-C1** does. However, it may produce overshrunk centerlines (see the sixth short “finger” in Fig. 2) compared to algorithm **6SI-C1** which, on the other hand, extracts skeletons containing more unwanted line segments (see the earless horse in Fig. 4). It is not surprising since endpoint characterization **C2** is more restrictive than **C1**. It can be seen that the existing algorithm **PK-C** produces several unwanted side branches that are not present in the centerlines of the new algorithms **6SI-C1** and **6SI-C2**.

Note that skeletonization is rather sensitive to coarse object boundaries. The false segments included by the produced skeletons must be removed by a pruning step [18].

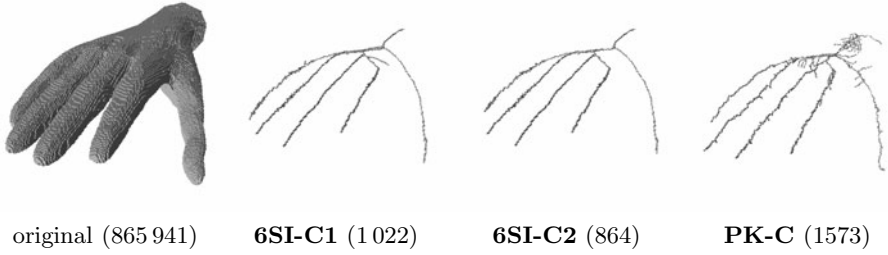


Fig. 2. A $174 \times 103 \times 300$ image of a hand and its centerlines produced by the three curve-thinning algorithms under comparison

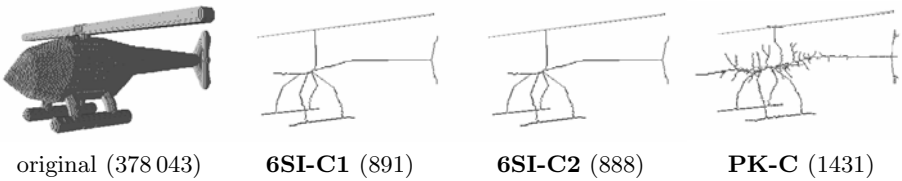


Fig. 3. A $304 \times 96 \times 261$ image of a helicopter and its centerlines produced by the three curve-thinning algorithms under comparison

The medial surfaces of the non-tubular test objects in Figs. 5-7 were extracted by the three surface-thinning algorithms **6SI-S1**, **6SI-S2**, and **GB-S**. Note that algorithm **6SI-S2** produces much less skeletal points than algorithm **6SI-S1** does: outer “corners” and “edges”, which remain connected with the inner skeletal parts, are not deleted by algorithm **6SI-S1**. It can be seen that the existing algorithm **GB-S** produces overshrunk seams between sheets.

For the test objects without any holes or cavities in Figs. 2, 4, and 6, our algorithm **6SI-TK** produces only one isolated point as their topological kernel (which is not depicted in Fig. 8). The topological kernels of the remaining test

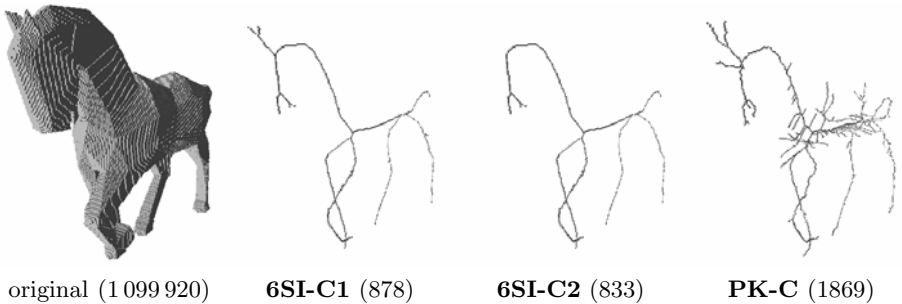


Fig. 4. A $300 \times 239 \times 83$ image of a horse and its centerlines produced by the three curve-thinning algorithms under comparison

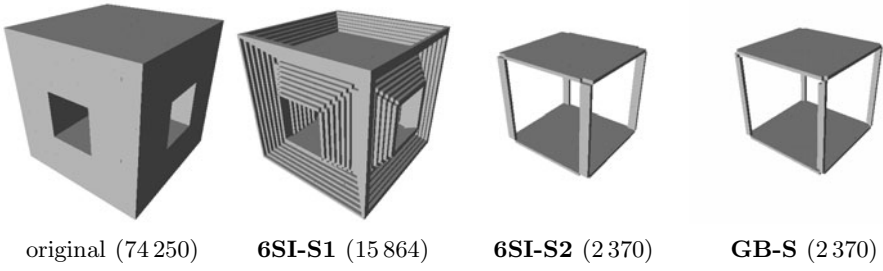


Fig. 5. A $45 \times 45 \times 45$ cube with two holes and its medial surfaces produced by the three surface-thinning algorithms under comparison

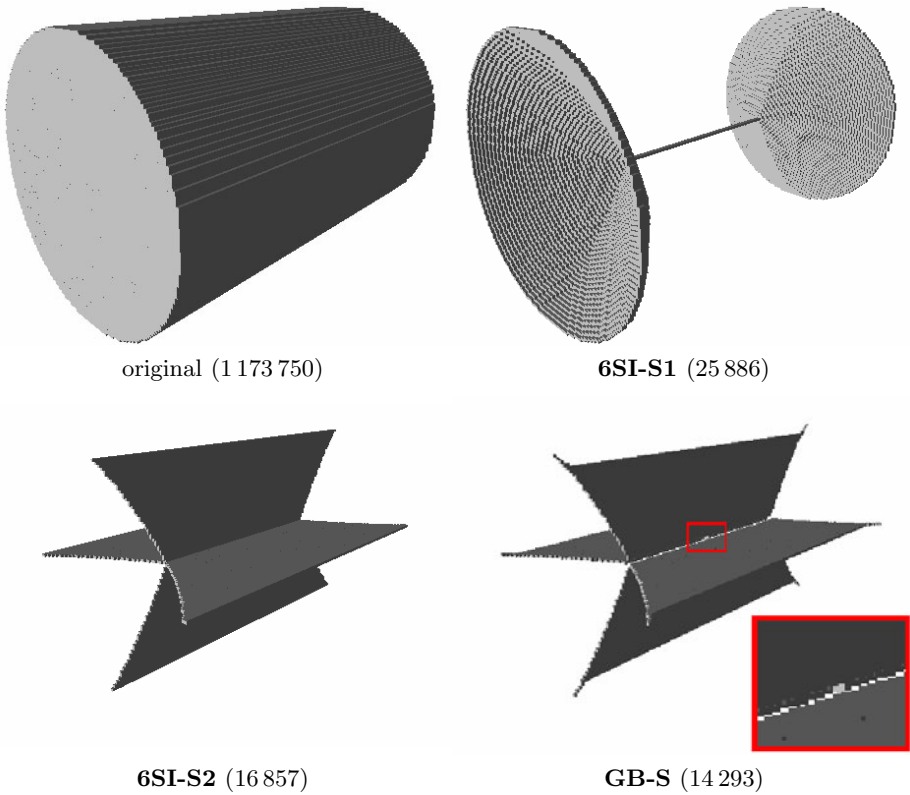


Fig. 6. A $104 \times 104 \times 152$ image of a cylinder and its medial surfaces produced by the three surface-thinning algorithms under comparison. Note that algorithm **GB-S** produced an overshrun seam between sheets.

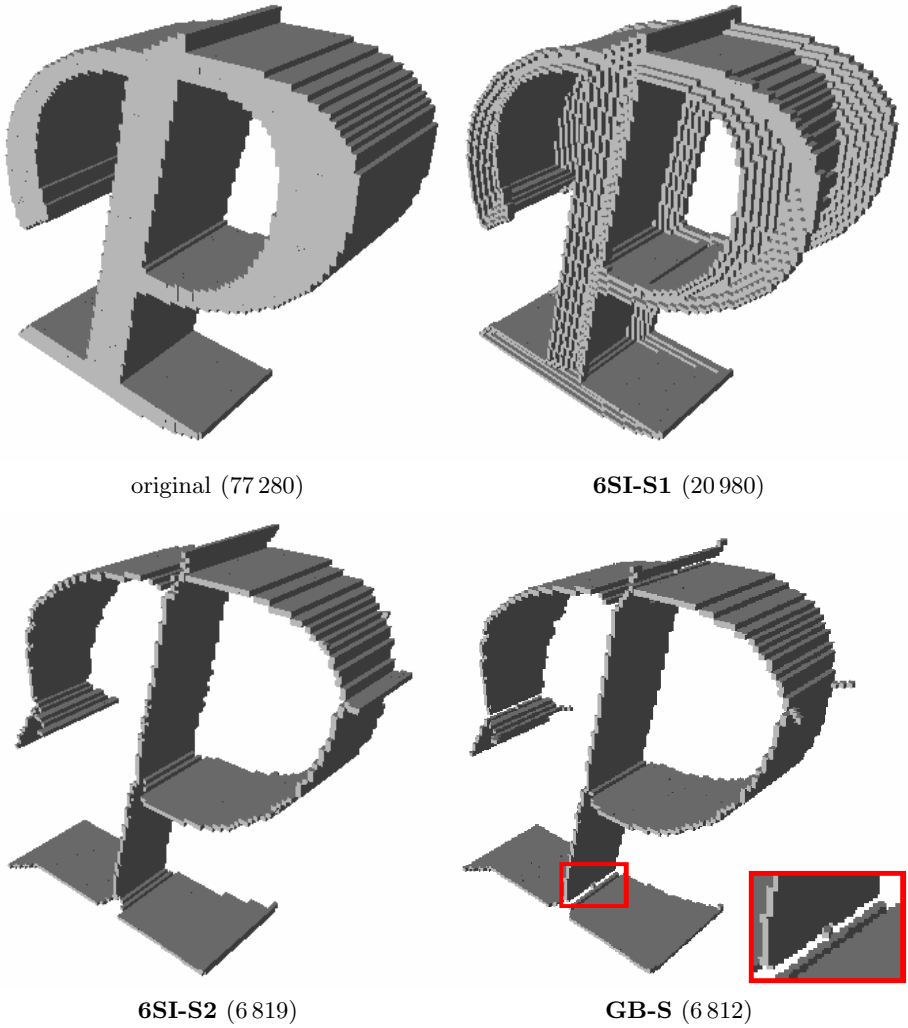


Fig. 7. A $100 \times 100 \times 30$ image of an object with a hole and its medial surfaces produced by the three surface-thinning algorithms under comparison. Note that algorithm **GB-S** produced an overshrunk seam between sheets.

objects containing some holes in Figs. 3, 5, and 7 are formed by 1-point wide closed curves (see Fig. 8).

By adapting the efficient implementation method presented in [16] our algorithms can be well applied in practice: they are capable of extracting skeleton-like features from large 3D shapes within one second on a usual PC.

The proposed implementation uses a pre-calculated look-up-table to encode the simple points. Since the simplicity of a point p can be decided by examining

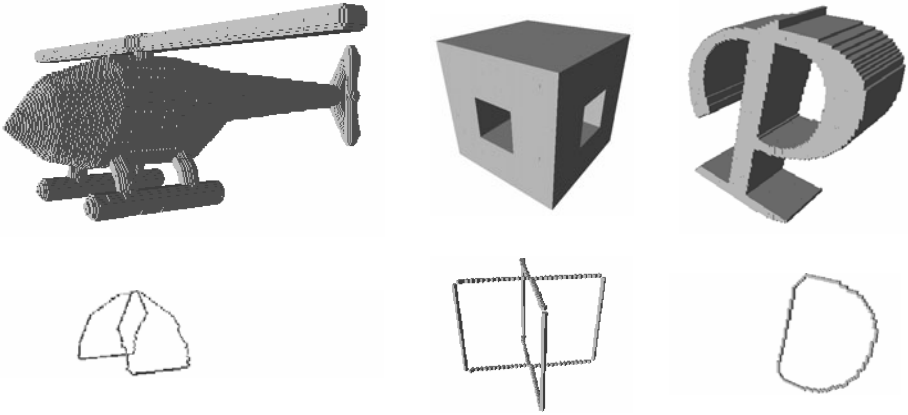


Fig. 8. Three objects with holes (upper row) and their topological kernels produced by algorithm **6SI-TK** (lower row). The extracted structures do not contain any simple points and they are topologically equivalent to the original objects.

the set $N_{26}(p)$, that look-up-table has 2^{26} entries of 1 bit in size, hence it requires just 8 MB of storage space in memory.

In addition, two lists/sets are used to speed up the process: the first one for storing the border points in the current picture. It is easy to see that thinning algorithms can only delete border-points, thus the repeated scans of the entire array storing the actual picture are not needed. The second list/set is to store all points that are “potentially deletable” in the current subiteration. At each phase of the thinning process, the deletable points are deleted, and the list of border points is updated accordingly.

The array storing the actual picture may contain five kinds of values: the value of “0” corresponds to “0” points, the value of “1” corresponds to interior points, the value of “2” is assigned to border points (that are stored in the first list/set), the value of “3” is assigned to all points that satisfy Condition 1 of Definition 4, and the value of “4” corresponds to all points that satisfy Conditions 1 and 2 of Definition 4.

6 Conclusions

Fast and reliable extraction of skeleton-like shape features (i.e., medial surface, centerline, and topological kernel) is extremely important in numerous applications for large 3D shapes. In this paper, we presented a new scheme for 6-subiteration parallel 3D thinning algorithms that is based on our new sufficient conditions for topology preservation. Hence the topological correctness of our algorithms is guaranteed. Five variations for the proposed thinning scheme were presented by considering five different characterizations of endpoints. Additional types of endpoints coupled with our general thinning scheme yield newer thinning algorithms.

Acknowledgements

This research was supported by the TÁMOP-4.2.2/08/1/2008-0008 program of the Hungarian National Development Agency, the European Union and the European Regional Development Fund under the grant agreement TÁMOP-4.2.1/B-09/1/KONV-2010-0005, and the grant CNK80370 of the National Office for Research and Technology (NKTH) & the Hungarian Scientific Research Fund (OTKA).

References

1. Arcelli, C., di Baja, G.S., Serino, L.: New removal operators for surface skeletonization. In: Kuba, A., Nyúl, L.G., Palágyi, K. (eds.) DGCI 2006. LNCS, vol. 4245, pp. 555–566. Springer, Heidelberg (2006)
2. Bertrand, G., Aktouf, Z.: A 3D thinning algorithms using subfields. In: Proc. SPIE Conf. on Vision Geometry III, vol. 2356, pp. 113–124 (1994)
3. Gong, W.X., Bertrand, G.: A simple parallel 3D thinning algorithm. In: Proc. 10th Int. Conf. on Pattern Recognition, pp. 188–190 (1990)
4. Hall, R.W.: Parallel connectivity-preserving thinning algorithms. In: Kong, T.Y., Rosenfeld, A. (eds.) Topological algorithms for digital image processing, pp. 145–179. Elsevier, Amsterdam (1996)
5. Kong, T.Y.: On topology preservation in 2-d and 3-d thinning. *Int. Journal of Pattern Recognition and Artificial Intelligence* 9, 813–844 (1995)
6. Kong, T.Y., Rosenfeld, A.: Digital topology: Introduction and survey. *Computer Vision, Graphics, and Image Processing* 48, 357–393 (1989)
7. Ma, C.M.: On topology preservation in 3D thinning. *CVGIP: Image Understanding* 59, 328–339 (1994)
8. Ma, C.M., Wan, S.Y.: A medial-surface oriented 3-d two-subfield thinning algorithm. *Pattern Recognition Letters* 22, 1439–1446 (2001)
9. Ma, C.M., Wan, S.Y., Chang, H.K.: Extracting medial curves on 3D images. *Pattern Recognition Letters* 23, 895–904 (2002)
10. Ma, C.M., Wan, S.Y., Lee, J.D.: Three-dimensional topology preserving reduction on the 4-subfields. *IEEE Trans. Pattern Analysis and Machine Intelligence* 24, 1594–1605 (2002)
11. Manzanera, A., Bernard, T.M., Pretéux, F., Longuet, B.: Medial faces from a concise 3D thinning algorithm. In: Proc.7th IEEE Int. Conf. Computer Vision, ICCV 1999, pp. 337–343 (1999)
12. Németh, G., Kardos, P., Palágyi, K.: Topology preserving 2-subfield 3D thinning algorithms. In: Proc. 7th IASTED Int. Conf. Signal Processing, Pattern Recognition and Applications, pp. 310–316 (2010)
13. Németh, G., Kardos, P., Palágyi, K.: Topology preserving 3D thinning algorithms using four and eight subfields. In: Campilho, A., Kamel, M. (eds.) ICIAR 2010. LNCS, vol. 6111, pp. 316–325. Springer, Heidelberg (2010)
14. Palágyi, K., Kuba, A.: A 3D 6-subiteration thinning algorithm for extracting medial lines. *Pattern Recognition Letters* 19, 613–627 (1998)
15. Palágyi, K., Kuba, A.: A parallel 3D 12-subiteration thinning algorithm. *Graphical Models and Image Processing* 61, 199–221 (1999)
16. Palágyi, K.: A 3D fully parallel surface-thinning algorithm. *Theoretical Computer Science* 406, 119–135 (2008)

17. Palágyi, K., Németh, G.: Fully parallel 3D thinning algorithms based on sufficient conditions for topology preservation. In: Brlek, S., Reutenauer, C., Provençal, X. (eds.) DGCI 2009. LNCS, vol. 5810, pp. 481–492. Springer, Heidelberg (2009)
18. Shaked, D., Bruckstein, A.: Pruning medial axes. *Computer Vision and Image Understanding* 69, 156–169 (1998)
19. Siddiqi, K., Pizer, S.: Medial representations – Mathematics, algorithms and applications. *Computational Imaging and Vision*, vol. 37. Springer, Heidelberg (2008)
20. Wang, T., Basu, A.: A note on ‘A fully parallel 3D thinning algorithm and its applications’. *Pattern Recognition Letters* 28, 501–506 (2007)

On Topology Preservation for Hexagonal Parallel Thinning Algorithms

Péter Kardos and Kálmán Palágyi

Department of Image Processing and Computer Graphics,
University of Szeged, Hungary
{pkardos,palagyik}@inf.u-szeged.hu

Abstract. Topology preservation is the key concept in parallel thinning algorithms on any sampling schemes. This paper establishes some sufficient conditions for parallel thinning algorithms working on hexagonal grids (or triangular lattices) to preserve topology. By these results, various thinning (and shrinking to a residue) algorithms can be verified. To illustrate the usefulness of our sufficient conditions, we propose a new parallel thinning algorithm and prove its topological correctness.

Keywords: hexagonal grids, parallel reduction operators, topology preservation, thinning.

1 Introduction

Thinning is an iterative layer-by-layer erosion until skeleton-like shape features of binary objects are left [2,5,11]. A thinning algorithm should preserve topology, that is, the produced output pictures should be topologically equivalent to the input ones for all possible digital binary pictures [4]. Parallel thinning algorithms are composed of parallel reduction operators (i.e., some object points having value of “1” in a binary picture that satisfy certain topological and geometric constraints are changed to “0” ones simultaneously) [2].

Sufficient conditions for topology preserving parallel reduction operators working on orthogonal grids have been given in 2D [2,3,8] and 3D [3,6]. These results provide methods of verifying that a parallel thinning (and shrinking to a residue) algorithm preserves topology.

Digital pictures on non-orthogonal grids have been studied by a number of authors [4,7]. A hexagonal grid is formed by a tessellation of regular hexagons. By duality, it corresponds to the triangular lattice, where the points are the centers of that hexagons, see Fig. 1. Hexagonal grids have a major advantage over the orthogonal ones. In 2D orthogonal/rectangular grids, the 8-adjacency relation is frequently used [4], where the length of diagonal moves is $\sqrt{2} \cdot a$ if the length of the horizontal and vertical moves is a . In hexagonal sampling scheme, each pixel is surrounded by six equidistant nearest neighbors. This results in a less ambiguous connectivity structure and in a better angular resolution compared to the rectangular case.

The majority of existing thinning algorithms work on orthogonal grids [2,11]. However, some parallel thinning methods were also proposed for hexagonal grids [1,9,10,12].

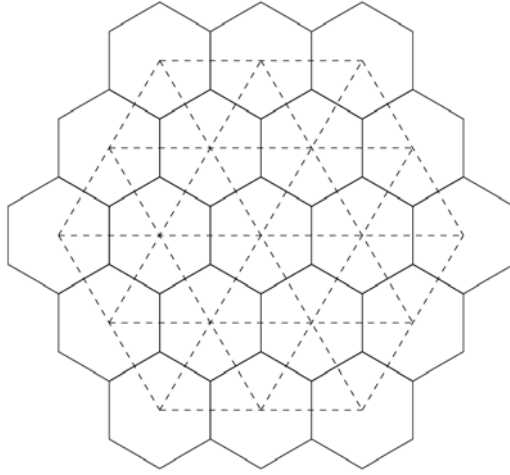


Fig. 1. A hexagonal grid and the corresponding triangular lattice

In this work we establish sufficient conditions for topology preserving parallel reduction operators in binary digital images sampled on hexagonal grids. By our results, various thinning (and shrinking to a residue) algorithms can be proved to be topology preserving.

The rest of this paper is organized as follows. Section 2 summarizes the basic notions of 2D digital topology. In Section 3, we discuss the mentioned sufficient conditions. To illustrate the usefulness of these conditions, we propose a new parallel subiteration-based thinning algorithm in Section 4 and prove that it is topology preserving for $(6, 6)$ pictures.

2 Basic Notions

Let us consider a hexagonal grid denoted by H , and let p be a pixel in H . Let us denote $N_6(p)$ the set of pixels being *6-adjacent* to pixel p and let $N_6^*(p) = N_6(p) \setminus \{p\}$. Figure 2 shows the 6-neighbors of a point p denoted by $N_6(p)$. The pixel denoted by p_i is called as the *i -th neighbor* of the central pixel p .

The sequence S of distinct pixels $\langle x_0, x_1, \dots, x_n \rangle$ is called a *6-path* of length n from pixel x_0 to pixel x_n in a non-empty set of pixels X if each pixel of the sequence is in X and x_i is 6-adjacent to x_{i-1} for each $1 \leq i \leq n$. Note that a single pixel is a 6-path of length 0. In the special case when $x_0 = x_n$ in S , we talk about a *6-cycle*, and the sequence $\langle x_i, x_{i+1}, \dots, x_j \rangle$ ($0 \leq i \leq j \leq n$) is called a *subpath of S* . Two pixels are said to be *6-connected* in set X if there is a 6-path in X between them.

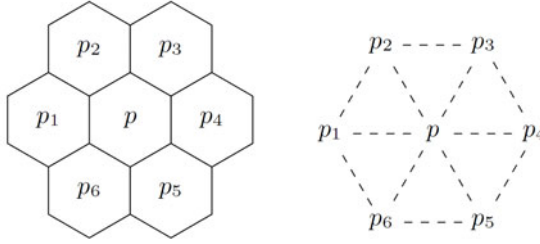


Fig. 2. Indexing scheme for the elements of $N_6(p)$ on hexagonal grid (left) and triangular lattice (right)

Based on the concept of digital pictures as reviewed in [4] we define the *2D binary (6,6) digital picture* as a quadruple $\mathcal{P} = (H, 6, 6, B)$. The elements of H are called the *pixels* of \mathcal{P} . Each pixel in $B \subseteq H$ is called a *black pixel* and has a value of 1. Each pixel in $H \setminus B$ is called a *white pixel* and the value of 0 is assigned to it. 6-adjacency is associated with both black and white pixels. A *black component* or an *object* is a maximal 6-connected set of pixels in B , while a *white component* or a *cavity* is a maximal 6-connected set of pixels in $H \setminus B$. An object composed of three mutually 6-adjacent black pixels is a *unit triangle*.

Let us denote $C_6(p)$ the number of black components in picture $(H, 6, 6, B \cap N_6^*(p))$. A black pixel is called a *border pixel* in a $(6, 6)$ picture if it is 6-adjacent to at least one white pixel. A black pixel p is called an *i -border pixel* in a $(6, 6)$ picture if its i -th neighbor (denoted by p_i in Fig. 2) is a white pixel ($1 \leq i \leq 6$). A black pixel p is called an *end pixel* in a $(6, 6)$ picture if it is 6-adjacent to exactly one black pixel.

A *reduction operator* transforms a binary picture only by changing some black pixels to white ones (which is referred to as the deletion of 1's). A *parallel reduction operator* deletes all pixels satisfying its condition simultaneously. A 2D reduction operator does *not* preserve topology [3] if any black component is split or is completely deleted, any white component is merged with another white component, or a new white component is created.

A *simple pixel* is a black pixel whose deletion is a topology preserving reduction [4]. Let \mathcal{P} be a $(6, 6)$ picture. The set of black pixels $D = \{d_1, \dots, d_k\}$ is called a *simple set* of \mathcal{P} if D can be arranged in a sequence $\langle d_{i_1}, \dots, d_{i_k} \rangle$ in which d_{i_1} is simple and each d_{i_j} is simple after $\{d_{i_1}, \dots, d_{i_{j-1}}\}$ is deleted from \mathcal{P} , for $j = 2, \dots, k$. (By definition, let the empty set be simple.)

3 Sufficient Conditions for Topology Preserving Parallel Reductions

In this section we discuss two important relationships for topology preservation in $(6, 6)$ pictures. We will prove in Theorem 1 that the simplicity of a pixel in a $(6, 6)$ picture is a local property, and based on this rule we will give sufficient conditions for a parallel reduction operator to preserve topology in Theorem 2.

Theorem 1. *Black pixel p in picture $(H, 6, 6, B)$ is simple if and only if both of the following conditions are satisfied:*

1. p is a border pixel.
2. $C_6(p) = 1$.

Proof. First we show indirectly that if Conditions 1 and 2 are satisfied, then p is simple. Let us suppose that the above conditions hold for p , and if we delete p , an object will be split into more components. Then there must be two pixels $q, r \in B$ such that all 6-paths in B between q and r contain p . This implies that all 6-paths in B between q and r contain a subpath $\langle p_i, p, p_j \rangle$ ($i, j \in \{1, 2, \dots, 6\}, i \neq j$), as well (see Fig. 3a). However, this contradicts Condition 2, by which p_i and p_j are also 6-connected in picture $(H, 6, 6, B \setminus \{p\})$. Hence, no object is split by the removal of p .

Now let us suppose that if we delete p then a white component is merged with another white component (or with the background). Then, there exist two white pixels $q, r \in H \setminus B$ such that all 6-paths in $(H \setminus B) \cup \{p\}$ between q and r contain p . Thus, all 6-paths in $H \setminus B$ from q to r contain a subpath $\langle p_i, p, p_j \rangle$ ($i, j \in \{1, 2, \dots, 6\}, i \neq j$), as well. Let us consider the case when $i = 1$. $j \neq 2$, or else the path $\langle p_i, p_j \rangle$ would be also a subpath of a 6-path from q to r , but this subpath does not contain p . If $j = 3$ (see Fig. 3b), then, according to our assumption, both of the 6-paths $\langle p_1, p_2, p_3 \rangle$ and $\langle p_3, p_4, p_5, p_6, p_1 \rangle$ must contain a black pixel. Similarly, if $j = 4$ (see Fig. 3c), then the 6-paths $\langle p_1, p_2, p_3, p_4 \rangle$ and $\langle p_4, p_5, p_6, p_1 \rangle$ must contain a black pixel. But for both of the possible cases we come into a contradiction with Condition 2. Because of the symmetry of the neighborhood of p composed by the elements of $N_6(p)$ (see Fig. 2), we can derive similar results if we change the values of the indexes i, j . Hence, no white component is merged with another white component (or with the background) by the removal of p .

If a new white component would be arisen when deleting p , then p could not be a border pixel in picture $(H, 6, 6, B)$, but this means a contradiction with Condition 1.

If an object would be deleted by the removal of p , then this would mean that p is an isolated object pixel, which implies $C_6(p) = 0$. However, this is a contradiction with Condition 2.

For all possible cases we came into a contradiction with Condition 1 or 2, therefore, p is a simple pixel. Now we will also indirectly show that if p is simple, then Conditions 1 and 2 hold.

Let us suppose that p is a simple pixel but at least one from the above mentioned conditions fails to hold. If p would not be a border pixel, then a new white component would be arisen by the removal of p , which can not happen because of the simplicity of p , therefore, $C_6(p) \neq 1$ must be satisfied.

As p is a simple pixel, the deletion of p does not lead to splitting an object into more components. This can only happen if between any two pixels p_i, p_j ($i, j \in \{1, 2, \dots, 6\}, i \neq j$) there exists a 6-path in B not containing p . If we add p to this path, then we obviously get a 6-cycle in B . Let us denote this 6-cycle

P_1 (see the continuous line in Fig. 31d). From the property $C_6(p) \neq 1$ follows that such a 6-cycle does not only contain pixels from $N_6^*(p)$, hence there exist such $q, r \in (H \setminus B) \cap N_6^*(p)$ for which the 6-path from p_i to p_j in set $N_6^*(p) \setminus \{r\}$ contains q and the 6-path from p_i to p_j in set $N_6^*(p) \setminus \{q\}$ contains r . There must be a 6-path between q and r in the set $H \setminus B$, or else a white component would be merged with another white component (or with the background) by the removal of p , which is not possible. If we add p to this path, we get a 6-cycle in $H \setminus B \cup \{p\}$ that we denote by P_2 (see the dotted line in Fig. 31d).

Let $S_{P_k} = \{x \mid x \text{ is contained in } P_k\}$ ($k \in \{1, 2\}$). It is easy to see that both of the pictures $(H, 6, 6, H \setminus S_{P_1})$ and $(H, 6, 6, H \setminus S_{P_2})$ contain exactly two objects. One of the objects in picture $(H, 6, 6, H \setminus S_{P_1})$ necessarily contains all elements of set $S_{P_2} \setminus \{p\}$, or else q and r would not be 6-connected in picture $(H, 6, 6, H \setminus B)$, hence the removal of p would reduce the number of white components. It is obvious that this object in $(H, 6, 6, H \setminus S_{P_1})$ fully contains one of the objects in $(H, 6, 6, H \setminus S_{P_2})$, which we will denote by O . p_i or p_j must be a member of O , or else q and r would be also 6-connected in picture $(H, 6, 6, (H \setminus B) \cap N_6^*(p))$, hence one of the 6-paths from p_i to p_j in $N_6^*(p)$ should contain both q and r , which contradicts to our assumptions on q, r . However, p_i and p_j are contained in cycle P_1 , which means, $p_i, p_j \in S_{P_1}$, therefore, none of these pixels may be a member of O . According to the derived contradiction, Conditions 1 and 2 hold for simple pixel p . \square

Lemma 1. *Let p and q two 6-adjacent simple pixels in picture $\mathcal{P} = (H, 6, 6, B)$. The following statements are equivalent:*

1. p is simple in picture $(H, 6, 6, B \setminus \{q\})$, or q is simple in picture $(H, 6, 6, B \setminus \{p\})$.
2. $N_6^*(p) \cap N_6^*(q) \cap (H \setminus B)$ contains exactly one element.

Proof. The first part of the proof will be carried out indirectly. Let us suppose that Statement 1 is true but the set $S = N_6^*(p) \cap N_6^*(q) \cap (H \setminus B)$ contains two elements or $S = \emptyset$. First, let us examine the case when S contains two elements, i.e., both common 6-neighbors of p and q are white in picture \mathcal{P} . As both p and q are simple in this picture, $C_6(p) = C_6(q) = 1$ holds by Theorem 1, which is possible only if $N_6^*(p) \cap B = \{q\}$ and $N_6^*(q) \cap B = \{p\}$. But in this case, p is an isolated object pixel in picture $(H, 6, 6, B \setminus \{q\})$, thus the removal of p would also result in the removal of an object. Hence, p is not a simple pixel in this case, which contradicts the condition on p . As a conclusion, $S = \emptyset$, i.e., both pixels in $N_6^*(p) \cap N_6^*(q)$ must be black. Let us denote these pixels by r_1 and r_2 . By Statement 1 and Theorem 1, $C_6(p) = 1$ in picture $(H, 6, 6, B \setminus \{q\})$ or $C_6(q) = 1$ in picture $(H, 6, 6, B \setminus \{p\})$. Because of the symmetry of the image part covered by the pixels of $N_6^*(p) \cap N_6^*(q)$, it is sufficient to only examine the first of these possible situations. In this case, there must be a 6-path from r_1 and r_2 in $N_6^*(p) \cap (B \setminus \{q\})$. This path necessarily contains all elements of $N_6^*(p) \setminus \{q\}$, from which follows that all 6-neighbors of p is black in picture \mathcal{P} . But in this case, p can not be a border pixel, which is a contradiction with our initial assumption. Hence, if Statement 1 is satisfied, then Statement 2 must be also fulfilled.

Now let us suppose that Statement 2 is true. We show that p is simple in picture $(H, 6, 6, B \setminus \{q\})$. As one of the 6-neighbors of q contained in $N_6^*(p)$ is white in picture \mathcal{P} , the removal of q from \mathcal{P} does not result in splitting of any component in picture $(H, 6, 6, B \cap N_6^*(p))$. Furthermore, as one of the 6-neighbors of q contained in $N_6^*(p)$ is black, the removal of q from \mathcal{P} does not lead to the removal of any object from the latter picture. Hence, $C_6(p) = 1$ still holds, which means, Statement 1 is also satisfied by Theorem [1](#). \square

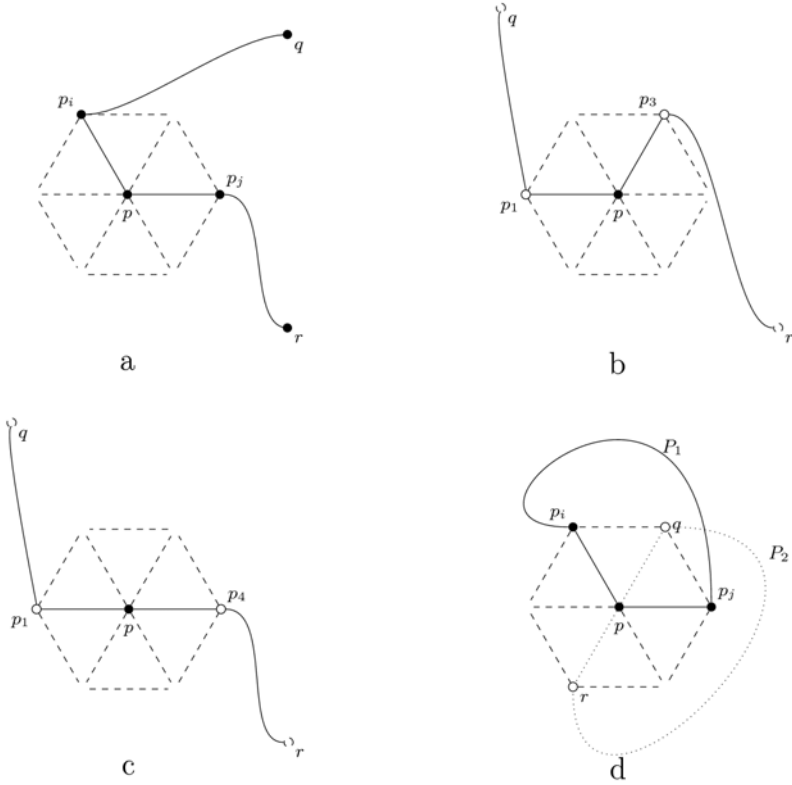


Fig. 3. The examined cases in the proof of Theorem [1](#)

Lemma 2. *Let \mathcal{O} be a parallel reduction operator, and let S be the set of black pixels removed by \mathcal{O} from an arbitrary picture $\mathcal{P} = (H, 6, 6, B)$. \mathcal{O} is topology-preserving, if for any pixel $p \in S$ and for any set $Q \subseteq S \cap N_6^*(p)$, p is simple in picture $(H, 6, 6, B \setminus Q)$.*

Proof. Let us suppose that the condition in the lemma is fulfilled on \mathcal{O} . As $\emptyset \subseteq S \cap N_6^*(p)$, any pixel $p \in S$ is simple in \mathcal{P} . Let $S = S_n = \{s_1, s_2, \dots, s_n\}$ ($n \in \mathbb{N}^+$), furthermore, let $S_i = \{s_1, s_2, \dots, s_i\}$ ($1 \leq i \leq n$). It is sufficient to see that S_n is a simple set in \mathcal{P} .

The proof will be done by induction on i . s_1 is simple in \mathcal{P} , hence set $S_1 = \{s_1\}$ is also simple in \mathcal{P} . Let us suppose that set S_k is simple in \mathcal{P} ($1 \leq k < n$). $S_{k+1} = S_k \cup \{s_{k+1}\}$, thus we have to prove that s_{k+1} is simple in picture $(H, 6, 6, B \setminus S_k)$.

It is useful to make a distinction between the pixels in S_k being and not-being 6-neighbors of s_{k+1} , as by Theorem [1](#), only the deletion of pixels in $N_6^*(s_{k+1})$ may influence the simplicity of s_{k+1} . Therefore, s_{k+1} remains simple in picture $\mathcal{P}_k = (H, 6, 6, B \setminus (S_k \setminus N_6^*(s_{k+1})))$. We only have to examine what happens if we remove some black 6-neighbors of s_{k+1} in \mathcal{P}_k . Let $Q_k = S_k \cap N_6^*(s_{k+1})$ (i.e., Q_k contains exactly that pixels of S_k which are 6-neighbors of s_{k+1}). As $S_k \subseteq S$, $Q_k = S_k \cap N_6^*(s_{k+1}) \subseteq S \cap N_6^*(s_{k+1})$ holds, too. It is easy to see that if we remove Q_k from the set of black pixels in \mathcal{P}_k , then we obtain the reduced set $B \setminus S_k$. Thus, if we apply the condition on \mathcal{O} for picture \mathcal{P}_k , we get that s_{k+1} is simple in picture $(H, 6, 6, B \setminus S_k)$. \square

Theorem 2. *A parallel reduction operator \mathcal{O} is topology-preserving in picture $\mathcal{P} = (H, 6, 6, B)$, if all of the following conditions hold:*

1. *Only simple pixels are deleted by \mathcal{O} .*
2. *If \mathcal{O} removes two 6-adjacent pixels p, q , then p is simple in picture $(H, 6, 6, B \setminus \{q\})$, or q is simple in picture $(H, 6, 6, B \setminus \{p\})$ (i.e., $\{p, q\}$ is a simple set).*
3. *\mathcal{O} does not delete completely any black component contained in a unit triangle.*

Proof. Let us suppose that \mathcal{O} satisfies Conditions 1-3, and let us denote S the set of black pixels deleted by \mathcal{O} . By Lemma [2](#) it is sufficient to show that, for any $p \in S$ and for any $Q \subseteq S \cap N_6^*(p)$, p is simple in picture $(H, 6, 6, B \setminus Q)$. This is obviously satisfied for $Q = \emptyset$ by Condition 1, thus we have only to examine the case when $Q \neq \emptyset$.

If $N_6^*(p) \cap S = \{q\}$, i.e., \mathcal{O} deletes exactly 1 black pixel from the 6-neighbors of p , then $Q = \{q\}$ must hold, and according to Conditions 1 and 2, p is simple in picture $(H, 6, 6, B \setminus Q)$.

Now let us assume that $N_6^*(p) \cap S = \{q, r\}$, i.e., \mathcal{O} deletes exactly 2 black pixels from the 6-neighbors of p . If Q contains only one element, then we can show similarly to the previous case that p is simple in picture $(H, 6, 6, B \setminus Q)$. Let us suppose that $Q = \{q, r\}$. The following two cases can be distinguished:

- I. q and r are 6-adjacent.
- II. q and r are not 6-adjacent.

First, let us examine Case I. By Conditions 1-2 and by Lemma [1](#), set $N_6^*(p) \cap N_6^*(r) \cap (H \setminus B)$ contains exactly one element, hence the 6-neighbor of r not coinciding with q is white in picture $(H, 6, 6, N_6^*(p) \cap B)$. It can be similarly derived that the 6-neighbor of q not coinciding with r is white in picture $(H, 6, 6, N_6^*(p) \cap B)$. Thus, set $\{q, r\}$ is a black component in picture $(H, 6, 6, N_6^*(p) \cap B)$. Because of the symmetrical arrangement of p, q, r , we can similarly prove that set $\{p, q\}$ is a black component in picture $(H, 6, 6, N_6^*(r) \cap B)$, and the set

$\{p, r\}$ is a black component in picture $(H, 6, 6, N_6^*(q) \cap B)$. As by Theorem [1](#), $C_6(p) = C_6(q) = C_6(r) = 1$ holds for \mathcal{P} , it also follows from the above conclusions that set $\{p, q, r\}$ is not 6-connected with any pixel $s \in B \setminus (N_6(p) \cup N_6(q) \cup N_6(r))$ in B . Therefore, set $\{p, q, r\}$ is surrounded only by white pixels in \mathcal{P} , which means that $\{p, q, r\}$ is a black component in \mathcal{P} . However, this contradicts to Condition 3, hence Case I can not come into question.

Now, let us discuss Case 2. According to Conditions 1-2 and Lemma [1](#), both of the sets $N_6^*(p) \cap N_6^*(q) \cap (H \setminus B)$ and $N_6^*(p) \cap N_6^*(r) \cap (H \setminus B)$ contain exactly one element. Obviously, $N_6^*(p) \cap N_6^*(r)$ contains two elements. From these also follows that set $N_6^*(p) \cap N_6^*(r) \cap B$ contains exactly one element. Let $N_6^*(p) \cap N_6^*(r) \cap B = \{s\}$. Based on Conditions 1 and 2, p is simple in picture $(H, 6, 6, B \setminus \{q\})$. Furthermore, by Condition 1, r is simple in \mathcal{P} , thus $C_6(r) = 1$ holds by Theorem [1](#) and as $q \notin N_6(r)$, $C_6(r) = 1$ still holds after the removal of q , which means that r is simple in picture $(H, 6, 6, B \setminus \{q\})$. By examining the possible arrangements of q and r , we can conclude that $q \neq s$, therefore the set $N_6^*(p) \cap N_6^*(r) \cap (H \setminus (B \setminus \{q\}))$ also contains exactly one element. Hence, by Lemma [1](#), p is simple in picture $(H, 6, 6, B \setminus Q)$.

Finally, we indirectly prove that \mathcal{O} may not delete more than 2 black 6-neighbors of p . Let $q_1, q_2, q_3 \in N_6^*(p)$ and let us suppose that \mathcal{O} deletes all the pixels of set $S = \{p, q_1, q_2, q_3\}$. If a pixel q_i ($i \in \{1, 2, 3\}$) would be 6-adjacent to every element of $S \setminus \{q_i\}$, then $N_6(p) \cap N_6(q_i) \cap (H \setminus B) = \emptyset$ would hold. However, this would lead to a contradiction with Lemma [1](#), hence this case is not possible. In every other cases, it is sure that some pixels q_i and q_j ($i, j \in \{1, 2, 3\}, i \neq j$) are not 6-connected in picture $(H, 6, 6, B \cap N_6^*(p))$, which implies that $C_6(p) > 1$. But this is not possible as p is a simple pixel in \mathcal{P} , which means by Theorem [1](#) that $C_6(p) = 1$ holds, thus we came into a contradiction again. \square

4 A New Topology Preserving Thinning Algorithm

In this section we introduce a new parallel thinning algorithm working on hexagonal arrays. The strategy which is used is called subiteration-based [2](#): each iteration step is composed of six parallel reduction operators according to the six directions assigned to the six neighbors of a pixel in a hexagonal array. Deletable pixels assigned to the i -th subiterations are given by matching template T_i , see Fig. [4](#) ($1 \leq i \leq 6$).

Algorithm 1 outlines the proposed 6-subiteration parallel thinning algorithm.

It is easy to see that Algorithm 1 cannot delete any end pixel since each template T_i ($1 \leq i \leq 6$) matches black pixels that are 6-adjacent to more than one black pixels (see Fig. [4](#)). Hence Algorithm 1 can produce medial curves.

In experiments Algorithm 1 was tested on objects of different shapes. Figure [5](#) presents three illustrative examples.

Now we show that the proposed algorithm is topology preserving. At first, some important properties of pixels that are matched by template T_1 are stated.

Proposition 1. *All pixels deleted by the parallel reduction operator given by template T_1 are 1-border pixels.*

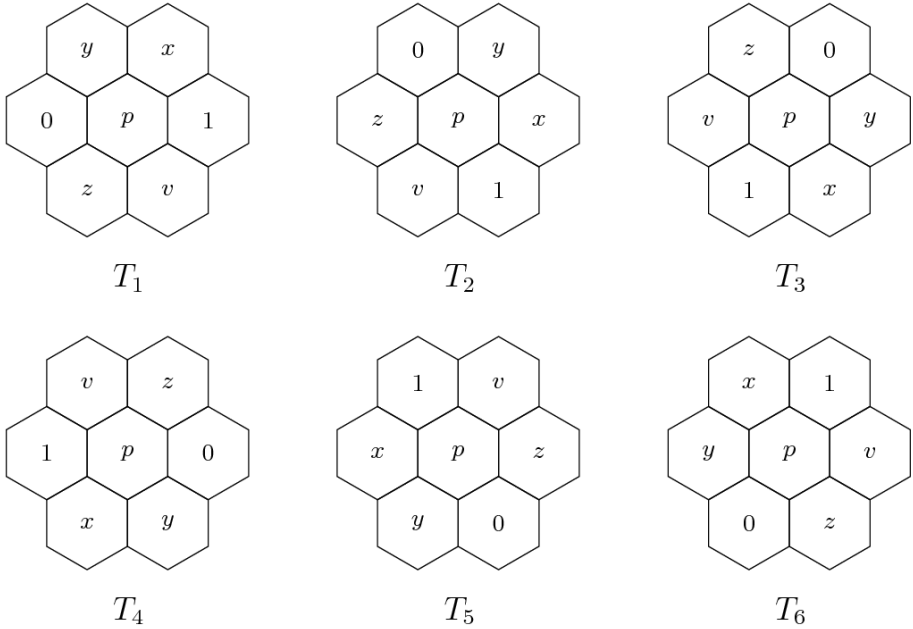


Fig. 4. Matching templates of the proposed subiteration-based algorithm. Template T_i is assigned to the i -th subiteration ($i = 1, 2, \dots, 6$). Notations (for each template): positions marked “p” and “1” match two black pixels; position marked “0” matches a white pixel; at least one positions marked “x” and “v” matches a black pixel; if position “y” matches a black pixel, then position “x” matches a black pixel, too; if position “z” matches a black pixel, then position “v” matches a black pixel, as well.

This holds since the 1st neighbor of a black pixel matched by T_1 is a white pixel.

Proposition 2. *If a black pixel can be deleted by template T_1 , then its 4th neighbor cannot be deleted by T_1 .*

Algorithm 1

Input: picture $(H, 6, 6, X)$

Output: picture $(H, 6, 6, Y)$

$Y = X$

repeat

 // one iteration step

for $i = 1$ **to** 6 **do**

 // subiteration for deleting some i -border pixels simultaneously

$D(i) = \{ p \mid p \in Y \text{ is matched by template } T_i \}$

$Y = Y \setminus D(i)$

until $D(1) \cup \dots \cup D(6) = \emptyset$

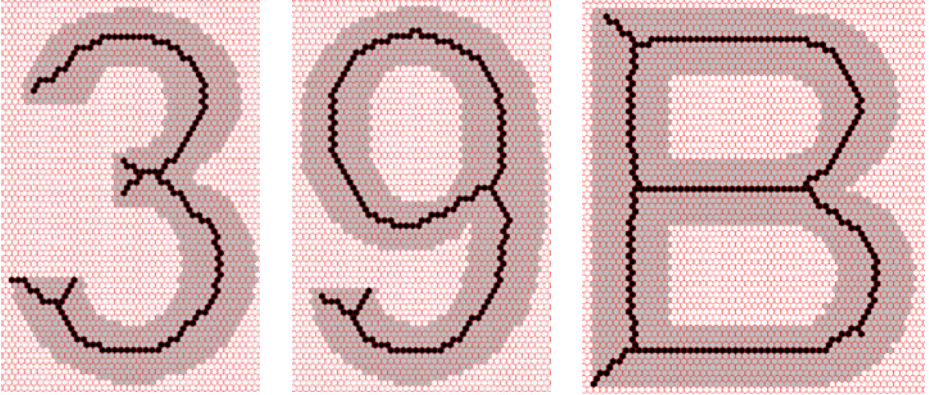


Fig. 5. Thinning of three objects sampled on hexagonal grids. Medial curves produced by Algorithm 1 are superimposed on the original objects.

This holds by Proposition 1, since if an object pixel matches template T_1 , then the 4th neighbor of that pixel can not be a 1-border pixel, hence that neighbor does not match T_1 .

Theorem 3. *Algorithm 1 is topology preserving for (6, 6) pictures.*

Proof. To prove it, we show that the parallel reduction operator given by template T_1 satisfies all conditions of Theorem 2.

1. Let us examine the simplicity of a pixel p that is matched by template T_1 . The first thing we need to verify that p is a border pixel. This holds by Proposition 1, hence Condition 1 of Theorem 1 is satisfied. To prove that Condition 2 of Theorem 1 holds, we show that $C_6(p) = 1$ for any p deleted by T_1 . This is fulfilled since if position “ y ” matches a black pixel, then position “ x ” matches a black pixel, and if position “ z ” matches a black pixel, then position “ v ” matches a black pixel (see Fig. 4). Hence Condition 1 of Theorem 2 is satisfied.
2. It was proved that only simple pixels are deleted by T_1 . To prove that Condition 2 of Theorem 2 holds, we show that for each pixel p deleted by T_1 , $C(p)$ remains 1 after a neighbor of p is deleted by T_1 .
 - The 1st neighbor of p is a white pixel.
 - The 2nd neighbor of p coincides with the template position “ y ”. If it can be deleted by T_1 , then the template position “ x ” matches a black pixel. Hence $C(p) = 1$ after the deletion of its 2nd neighbor.
 - The 3rd neighbor of p coincides with the template position “ x ”. If it can be deleted by T_1 , then template position “ y ” coincides with a white pixel. Hence $C(p) = 1$ after the deletion of its 3rd neighbor.
 - The 4th neighbor of p is a black pixel. Since p is its 1st neighbor, it cannot be deleted by T_1 .

- The 5th neighbor of p coincides with the template position “ v ”. If it can be deleted by T_1 , then template position “ z ” coincides with a white pixel. Hence $C(p) = 1$ after the deletion of its 5th neighbor.
- The 6th neighbor of p coincides with the template position “ z ”. If it can be deleted by T_1 , then the template position “ v ” matches a black pixel. Hence $C(p) = 1$ after the deletion of its 6th neighbor.

If pixel p is deleted by T_1 , then p is a 1-border pixel by Proposition 1. It is obvious that pixel p remains a 1-border pixel after $q \in N_6^*(p)$ is deleted by T_1 . Since p is a 1-border pixel and $C(p) = 1$ after the deletion of q , p remains a simple pixel after q is deleted by T_1 . Hence Condition 2 of Theorem 2 is satisfied.

3. Suppose that pixel p is an element of an arbitrary black component. If p is deleted by T_1 , then its 4th neighbor q is a black and it cannot be deleted by Proposition 2. Since both pixels p and q are in the same black component, no black component can be deleted completely by T_1 . Hence Condition 3 of Theorem 2 is satisfied.

We proved that the parallel reduction operator given by template T_1 is topology preserving. The remaining five subiterations of Algorithm 1 given by templates T_2, T_3, T_4, T_5 , and T_6 are topology preserving operators since these templates can be obtained by rotations of template T_1 . Algorithm 1 is topology preserving since it is composed of topology preserving operators. \square

5 Conclusions

This paper presents a characterization of simple point in $(6, 6)$ digital pictures sampled on hexagonal grids (or triangular lattices) and establishes sufficient conditions for topology preserving parallel reduction operators working on $(6, 6)$ pictures. By our results, various thinning (and shrinking to a residue) algorithms can be proved to be topology preserving. To illustrate the usefulness of our sufficient conditions, we have proposed a new parallel subiteration-based thinning algorithm and we have proved its topological correctness.

Acknowledgements

This research was supported by the TÁMOP-4.2.2/08/1/2008-0008 program of the Hungarian National Development Agency, the European Union and the European Regional Development Fund under the grant agreement TÁMOP-4.2.1/B-09/1/KONV-2010-0005, and the grant CNK80370 of the National Office for Research and Technology (NKTH) & the Hungarian Scientific Research Fund (OTKA).

References

1. Deutsch, E.S.: Thinning algorithms on rectangular, hexagonal, and triangular arrays. *Communications of the ACM* 15, 827–837 (1972)
2. Hall, R.W.: Parallel connectivity-preserving thinning algorithms. In: Kong, T.Y., Rosenfeld, A. (eds.) *Topological algorithms for digital image processing*, pp. 145–179. Elsevier, Amsterdam (1996)
3. Kong, T.Y.: On topology preservation in 2-d and 3-d thinning. *Int. Journal of Pattern Recognition and Artificial Intelligence* 9, 813–844 (1995)
4. Kong, T.Y., Rosenfeld, A.: Digital topology: Introduction and survey. *Computer Vision, Graphics, and Image Processing* 48, 357–393 (1989)
5. Lam, L., Lee, S.-W., Suen, C.Y.: Thinning methodologies — A comprehensive survey. *IEEE Trans. Pattern Analysis and Machine Intelligence* 14, 869–885 (1992)
6. Ma, C.M.: On topology preservation in 3D thinning. *CVGIP: Image Understanding* 59, 328–339 (1994)
7. Marchand-Maillet, S., Sharaiha, Y.M.: *Binary digital image processing – A discrete approach*. Academic Press, London (2000)
8. Ronse, C.: Minimal test patterns for connectivity preservation in parallel thinning algorithms for binary digital images. *Discrete Applied Mathematics* 21, 67–79 (1988)
9. Staunton, R.C.: An Analysis of Hexagonal Thinning Algorithms and Skeletal Shape Representation. *Pattern Recognition* 29, 1131–1146 (1996)
10. Staunton, R.C.: One-pass parallel hexagonal thinning algorithm. In: *Proc. IEE 7th Int. Conf. Image Processing and Its Applications*, pp. 841–845 (1999)
11. Suen, C.Y., Wang, P.S.P. (eds.): *Thinning methodologies for pattern recognition*. Series in Machine Perception and Artificial Intelligence, vol. 8. World Scientific, Singapore (1994)
12. Wiederhold, P., Morales, S.: Thinning on quadratic, triangular, and hexagonal cell complexes. In: Brimkov, V.E., Barneva, R.P., Hauptman, H.A. (eds.) *IWCIA 2008*. LNCS, vol. 4958, pp. 13–25. Springer, Heidelberg (2008)

Accurate Curvature Estimation along Digital Contours with Maximal Digital Circular Arcs

Tristan Roussillon¹ and Jacques-Olivier Lachaud²

¹ Université de Lyon,
Université Lyon 2, LIRIS, UMR5205, 69676, France
`tristan.roussillon@liris.cnrs.fr`

² LAMA, UMR CNRS 5127
Université de Savoie, Le Bourget-du-Lac, 73376, France
`jacques-olivier.lachaud@univ-savoie.fr`

Abstract. We propose in this paper a new curvature estimator based on the set of maximal digital circular arcs. For strictly convex shapes with continuous curvature fields digitized on a grid of step h , we show that this estimator is multigrad convergent if the discrete length of the maximal digital circular arcs grows in $\Omega(h^{-\frac{1}{2}})$. We indeed observed this order of magnitude. Moreover, experiments showed that our estimator is at least as fast to compute as existing estimators and more accurate even at low resolution.

1 Introduction

The accurate estimation of geometric parameters such as perimeter, tangent and curvature along digital objects is important for many image analysis applications, for instance the detection of dominant points and corners. We focus here on curvature estimation along digital contours, assumed to be digitizations of smooth Euclidean shapes of the plane. This subject has led to the development of many estimators, which fall roughly in three categories according to [17,16,8]: (i) derivative of the tangent orientation, (ii) norm of the second derivative of the curve considered as a path, (iii) inverse of the osculating circle radius. In most approaches, a user-given window or smoothing parameter is used so as to remove the jaggedness of digital contours and to make it continuous [17,16,4,12,13,3,5,6].

In approaches of (i), the curvature estimation relies on the convolution of the contour tangent by some derivative of Gaussian kernel, either in a continuous setting [17,16,4], or in a discrete setting [13,3,5]. Furthermore a preprocessing with digital straight segments is sometimes used to limit the arithmetic effects [16,4]. Methods of (ii) generally reconstruct locally the contour with some polynomial of given degree [14,8]. Again, an important parameter is the size of the window. Methods of (iii) tries to estimate the osculating circle around the point of interest [11,2,8,6].

Among all these curvature estimators, few do not require an external parameter. They all rely on digital straight segment (DSS) recognition. Only the length of DSS is used in [1] to estimate the osculating circle, but this approach

does not give accurate estimations. In [2] (HK2005 in [8]), the two half-tangents define a triangle whose circumscribed circle approximates the osculating circle. This technique gives correct results on average, but oscillates a lot and leads to very large errors as reported in [9]. The GMC estimator [9] computes, among all Euclidean shapes that are digitized as the input digital shape, the shape that minimizes its squared curvature. Digital straight segments are used here as a preprocessing to make easier the optimization.

An important property that should have a discrete estimator is the *multigrid convergence* [10]. Indeed, at a given resolution, infinitely many shapes have the same digitization, which hampers the objective comparison of estimators. For estimators of local geometric quantities like tangent or curvature, few results exist. We may quote some convergence results for tangent estimators [11,13,3]. And there is no correct convergence results for curvature as far as we know.

In this paper, we present in Section 2 a new curvature estimator based on Maximal Digital Circular Arcs detection (MDCA). It is thus a natural extension of uniformly convergent tangent estimators based on maximal DSS [11]. It is a parameter-free method, with linear computation time in practice. Section 3 discuss the multigrid convergence of this estimator and establishes the importance of the asymptotic length evolution of MDCAs as the resolution gets finer. Section 4 verifies experimentally the asymptotic behavior of MDCAs and of the curvature field estimation. These results backs up our claim that this estimator is multigrid-convergent. Section 5 compares numerically this estimator with the latest curvature estimators of the literature: the binomial convolution (BC) estimator of [13,3], and the global min-curvature (GMC) estimator of [9]. They show that this new estimator outperforms the previous ones on tested shapes.

2 Curvature Estimation Based on the Set of Maximal Digital Circular Arcs

We propose in this section a curvature estimator based on a curvature map computed from the set of maximal digital circular arcs.

Let \mathbb{X} be a family of compact simply connected subsets of \mathbb{R}^2 with continuous curvature fields. The reason that explains why we need continuous curvature fields is postponed to section 3.1. We denote by $D_h(X)$ the Gauss digitization of $X \in \mathbb{X}$ with grid step h , seen as a union of pixels of side h in \mathbb{R}^2 . For sake of clarity, we shorten in the sequel $D_h(X)$ into D and denote its complementary by \bar{D} . Moreover, let us assume that D contains at least one pixel, i.e. $|D| \geq 1$.

Let the *digital contour* C of D be the topological border ∂D of D . Any side of any pixel is a *grid edge*. The contour C is a circular list of grid edges in the clockwise orientation. Any part C' of C is a sequence of consecutive grid edges. The *discrete length* of C' is defined as its number of grid edges. The distance between two grid edges is defined as the discrete length of the shortest part joining these two grid edges. Each grid edge lies between two pixels, one belonging to D , the other belonging to \bar{D} . The centers of the pixels incident to C' and belonging to D are the *interior points* of C' , whereas those of the pixels incident to C' and belonging to \bar{D} are the *exterior points* of C' .

Any part C' of C is a *digital circular arc* (DCA for short) if and only if the interior and exterior points of C' are circularly separable, i.e. there exists a (Euclidean) circle that either encloses the interior points without enclosing any exterior points or that encloses the exterior points without enclosing any interior points. Given a grid step h , the map associating to any DCA A the value 0 if the interior and exterior points of A are linearly separable and the curvature of an arbitrary separating circle otherwise is denoted by k^h , such that $k^h(A)$ is the curvature of the part A (and h is the unit).

Any DCA A is maximal if and only if all the parts C' containing A , i.e. such that $A \subset C'$, are not a DCA. The set of all maximal DCAs (MDCAs for short) that lie on a given contour is unique. The first grid edge of any two distinct MDCAs cannot be identical because if it is, the shortest MDCA is necessarily contained in the longest one and is thus not maximal. Consequently, the MDCAs can be ordered according to the position of their first grid edge in the contour. Let us then denote by $(A_i)_{i \in 1, \dots, n}$ the sequence of the n MDCAs lying on C .

Given any DCA A of discrete length L , the map associating A to its middle grid edge is denoted by m such that $m(A)$ is the $\lfloor \frac{L}{2} \rfloor$ -th grid edge of A . Note that the middle grid edges of any two consecutive MDCA are never the same grid edge.

As a result, a contour C can be partitioned without ambiguity into a sequence $(V_i)_{i \in 1, \dots, n}$ such that V_i is the set of grid edges closer to $m(A_i)$ than to any other grid edge $m(A_j)$, $j \in 1, \dots, n$ and $j \neq i$ (the first one with respect to the clockwise orientation of the contour is assumed to be closer in case of tie). For all $i \in 1, \dots, n$, we associate to any grid edge of V_i , the curvature value of the separating circle associated to the MDCA A_i .

Definition 1. Let $(A_i)_{i \in 1, \dots, n}$ be the sequence of MDCAs lying on C . Let $(V_i)_{i \in 1, \dots, n}$ be a partition of C such that V_i is the set of grid edges closer to $m(A_i)$ than to any other grid edge $m(A_j)$, $j \in 1, \dots, n$ and $j \neq i$. Given a grid step h , the curvature estimator $\hat{\kappa}_{MDCA}^h$ is the piecewise constant function that associates to any digital contour C and to any point p of C a curvature value in \mathbb{R} such that:

$$\forall i \in 1, \dots, n, \forall e \in V_i, \forall p \in e, \hat{\kappa}_{MDCA}^h(C, p) = k^h(A_i).$$

A minimal example is provided in Fig. [11](#).

3 On the Multigrid Convergence of the MDCA Estimator

In this section, we first propose a discrete curvature estimator definition and next discuss of the multigrid convergence properties of $\hat{\kappa}_{MDCA}$.

3.1 Multigrid Convergence Definition for a Curvature Estimator

The curvature is some function of the shape boundary. However, the contour of the shape digitization does not define the same domain. Therefore we cannot directly compare the true curvature function with the estimated curvature

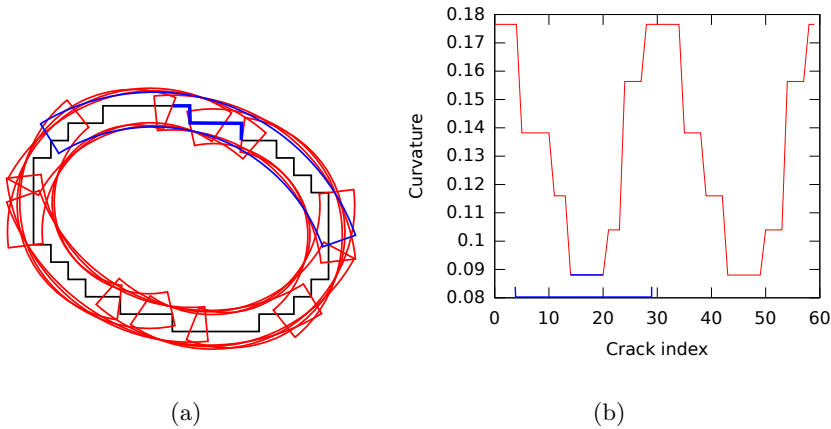


Fig. 1. The set of MDCAs (12 arcs) is depicted in a) with pieces of rings along the contour of the digitization of an ellipse having a great axis of 9 pixels long and a small axis of 6 pixels long. The angle between the main orientation and the x-axis is equal to 1.9 radians. The curvature plot defined from the set of MDCAs is shown in b). The blue grid edges are those whose curvature depends on the radius of the blue MDCA.

function. We provide below a definition of multigrid convergence for discrete curvature estimators. It is neither a parametric definition as in [3] nor a point-wise definition as the standard multigrid convergence reported in [10]. It is a geometric definition, stating that any digital point sufficiently close to the point of interest has its estimated curvature which tends toward the expected curvature. This definition of multigrid convergence imposes shapes with continuous curvature fields.

Let us recall that \mathbb{X} is a family of compact simply connected subsets of \mathbb{R}^2 with continuous curvature field. We denote by $D_h(X)$ the Gauss digitization of $X \in \mathbb{X}$ with grid step h . For any x in the topological boundary ∂X of X , let $\kappa(X, x)$ be the curvature of ∂X at x . A *discrete curvature estimator* $\hat{\kappa} = (\hat{\kappa}^h)_{h>0}$ is a family of mappings which associates to any digital contour C and a point $p \in C$ some value of \mathbb{R} . Note that the estimator proposed in Definition 1 is a discrete curvature estimator as defined above. The following section aims at studying the multigrid-convergence of this estimator:

Definition 2. *The estimator $\hat{\kappa}$ is multigrid-convergent for the family \mathbb{X} if and only if, for any $X \in \mathbb{X}$, $h > 0$, for any $x \in \partial X$,*

$$\forall y \in \partial D_h(X) \text{ with } \|y - x\|_1 \leq h, |\hat{\kappa}^h(D_h(X), y) - \kappa(X, x)| \leq \tau_x(h),$$

where $\tau_{X,x} : \mathbb{R}^{+*} \rightarrow \mathbb{R}^+$ has null limit at 0. This function defines the speed of convergence of $\hat{\kappa}$ toward κ at point x of X . The convergence is uniform for X

when every $\tau_{X,x}$ is bounded from above by a function τ_X independent of $x \in X$ with null limit at 0.

3.2 Relation with Growth of MDCAs

In this section, we establish a link between the multigrid convergence of our estimator and the asymptotic length of the MDCAs along the digital shape. We restrict our study to convex shapes having strictly positive and finite curvature.

We recall first that any convex shape X is uniquely determined by its *support function* f of center B , with $f : [0, 2\pi[\rightarrow \mathbb{R}$. In our case, $f(\phi)$ is the algebraic distance between B and the point of ∂X with normal $\mathbf{n}(\phi) = (\cos \phi, \sin \phi)$. A (θ_1, θ_2) -piece of (B, R, e) -ring is the subset of \mathbb{R}^2 defined in polar coordinates as $\{(r, \theta) : R - e \leq r \leq R + e, \theta_1 \leq \theta \leq \theta_2\}$. It is said to be *simply covering* ∂X if its intersection with ∂X is a simple curve whose extremities have a polar angle of respectively θ_1 and θ_2 .

Lemma 1. *Let \mathcal{R} be a (θ_1, θ_2) -piece of (B, R, h) -ring that simply covers ∂X , whose Euclidean length $R(\theta_2 - \theta_1)$ is lower bounded by $\Omega(h^a)$ and upper bounded by $O(h^b)$, $0 < b \leq a < 1/2$. Then, the radius R tends towards the inverse of the curvature of ∂X at any points of $\mathcal{R} \cap \partial X$ as $h \rightarrow 0$.*

Proof. Let \mathcal{R} be such a ring. Its length is $L = R(\theta_2 - \theta_1)$. The points of ∂X at θ_1 and θ_2 are respectively denoted by M_1 and M_2 . The points M_i , $i = 1, 2$, have normals $\mathbf{n}(\phi_i)$. We represent X with its support function f centered on B . We proceed in four steps.

1. We relate $\theta_2 - \theta_1$ and $\phi_2 - \phi_1$.

Let L_{12} be the length ∂X between M_1 and M_2 . By convexity of X , L_{12} is longer than the shortest arc of \mathcal{R} and shorter than the longest arc of \mathcal{R} plus twice its thickness. We get:

$$(R - h)(\theta_2 - \theta_1) \leq L_{12} \leq (R + h)(\theta_2 - \theta_1) + 4h$$

But $L_{12} = \int_{\phi_1}^{\phi_2} \frac{1}{\kappa} d\phi$. Introducing κ_{\min} and κ_{\max} as a lower and upper bound on the curvature in $\partial X \cap \mathcal{R}$, we easily get by integration:

$$\kappa_{\min}(R - h)(\theta_2 - \theta_1) \leq \phi_2 - \phi_1 \leq \kappa_{\max}((R + h)(\theta_2 - \theta_1) + 4h),$$

and since $L/R = \theta_2 - \theta_1$:

$$\kappa_{\min}(1 - h/R)L \leq \phi_2 - \phi_1 \leq \kappa_{\max}((1 + h/R)L + 4h), \tag{1}$$

2. We major f'' at point $\overline{M} \in \partial X$ whose normal has angle $\overline{\phi} = (\phi_1 + \phi_2)/2$. Finite differences applied on $f''(\overline{\phi})$ gives:

$$f''(\overline{\phi}) = \frac{f(\phi_1) - 2f(\overline{\phi}) + f(\phi_2)}{((\phi_2 - \phi_1)/2)^2} + O(\phi_2 - \phi_1). \tag{2}$$

Since M_1, \overline{M}, M_2 are all in \mathcal{R} and since the support function f has center B , the values $f(\phi_1), f(\overline{\phi}), f(\phi_2)$ are bounded by $R - h$ and $R + h$. We also insert (1) in (2) to get:

$$|f''(\overline{\phi})| \leq \frac{16h}{\kappa_{\min}^2 L^2} \left(\frac{1}{1 + O(h/R)} \right) + O(L(1 + h/R)) + O(h). \quad (3)$$

3. The quantity $f''(\overline{\phi})$ tends toward 0 as $h \rightarrow 0$.

The radius R of the ring is lower bounded by some constant for sufficiently small h since ∂X has finite curvature everywhere. As a result $h/R \rightarrow 0$ as $h \rightarrow 0$. We insert the hypothesis $\Omega(h^a) \leq L \leq O(h^b)$ into (3):

$$|f''(\overline{\phi})| \leq O(h^{1-2a}) + O(h^b) + O(h). \quad (4)$$

From $1/2 > a$ and $0 < b$, it is clear that $|f''(\overline{\phi})|$ tends toward 0 as $h \rightarrow 0$.

4. We relate the behavior of f'' to the curvature.

It is well known that the curvature has a close relation with the support function: $\forall \phi, 1/\kappa(\phi) = f(\phi) + f''(\phi)$. Now $\forall \phi, \phi_1 \leq \phi \leq \phi_2$, we have $R - h \leq f(\phi) \leq R + h$. This holds also for $\overline{\phi}$ which lies between ϕ_1 and ϕ_2 . We immediately obtain

$$1/\kappa(\overline{\phi}) = R + O(h) + O(f''(\overline{\phi})). \quad (5)$$

From (4) and (5), it is now obvious that the radius of the covering ring tends toward the inverse of the curvature of some point in $\mathcal{R} \cap \partial X$. The curvature of ∂X being continuous and \mathcal{R} being included in some ball of radius $O(h^b)$, $b > 0$, the same holds for all points of $\mathcal{R} \cap \partial X$. \square

It remains to show that there always exists a piece of ring simply covering ∂X for any MDCA.

Theorem 1. *Let \mathbb{X} be the family of compact convex subsets of \mathbb{R}^2 , whose curvature field is continuous, strictly positive and upper bounded. If the length of MDCA's along any $D_h(X)$, $X \in \mathbb{X}$, is lower bounded by $\Omega(h^a)$ and upper bounded by $O(h^b)$, $0 < b \leq a < 1/2$, then the curvature estimator $\hat{\kappa}_{MDCA}$ is uniformly multigrid convergent for X , with $\tau = O(h^{\min(1-2a,b)})$.*

Proof. Let $h > 0$, $x \in \partial X$, and $y \in \partial D_h(X)$, $\|y - x\|_1 \leq h$. The point y is closer to some MDCA A_i than to any other one. Let \mathcal{C} be any circle separating the interior and exterior points of A_i . Let us denote by B its center and R its radius. Let θ_1 and θ_2 be the polar angle (with respect to B) of the extremities of A_i . Let \mathcal{R} be the (θ_1, θ_2) -piece of $(B, R, \sqrt{2}h)$ -ring. See Fig. 2 for an illustration. The circle \mathcal{C} separates all the interior points from the exterior points of A_i . Since we have chosen the width $\sqrt{2}h$, we also know that all these points are included in \mathcal{R} . Since ∂X is contained in the one-pixel wide strip between interior and exterior points of the grid edges, ∂X can only exit \mathcal{R} on its sides. It is well known that for a sufficiently small h , the digital contour $\partial D_h(X)$ has the same topology as ∂X (e.g. par-regularity [7]). Therefore $\partial X \cap \mathcal{R}$ has the same topology as $\partial D_h(X) \cap \mathcal{R}$,

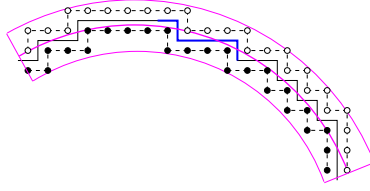


Fig. 2. Illustration of the proof of Theorem 1: piece of ring defined from the separating circle of a given MDCA

i.e. it is curve without intersection. We conclude that \mathcal{R} is simply covering ∂X . By definition \mathcal{R} covers y , but also x : indeed, it is the most centered MDCA, and an MDCA is at least longer than three pixels. Applying Lemma 1, gives that the inverse of R tends toward the curvature at x , and more precisely in $O(h^{\min(1-2a,b)})$ according to (4). We conclude since $\hat{\kappa}_{MDCA}$ gives by definition $1/R$ at y . \square

If $N = 1/h$ is the resolution, The discrete length of MDCAs (length divided by h) must thus grow at speed faster than $\Omega(h^{-\frac{1}{2}})$, i.e. $\Omega(\sqrt{N})$, and smaller than $O(h^{-1})$, i.e. $O(N)$, so that the curvature estimator $\hat{\kappa}_{MDCA}$ is multigrid convergent.

4 Experimental Evaluation

We experimentally observed on digitizations of ellipses that the average discrete length of the MDCAs grows in $\Theta(h^{-\frac{1}{2}})$ as h tends towards 0 (Fig. 3a). According to the results of the previous section, this suggests that the proposed estimator is multigrid convergent for strictly convex shapes having continuous curvature fields such as ellipses.

In order to observe the multigrid convergence of our estimator for an ellipse E of center c , we compared the value of $\hat{\kappa}_{MDCA}^h$ to the ground-truth $\hat{\kappa}$. For each grid edge, we associated to the midpoint p , its projection p' on ∂E along the ray coming from c so that the absolute error at p is defined as:

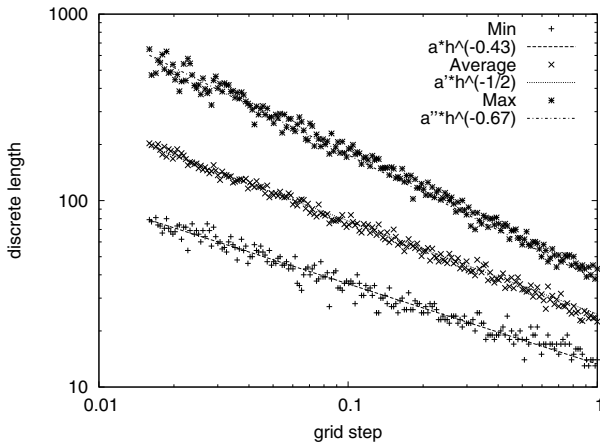
$$\epsilon_{abs}(p) = |\hat{\kappa}(\partial E, p') - \hat{\kappa}_{MDCA}^h(\partial D_h(E), p)| \quad (6)$$

The relative error at p is consequently defined as:

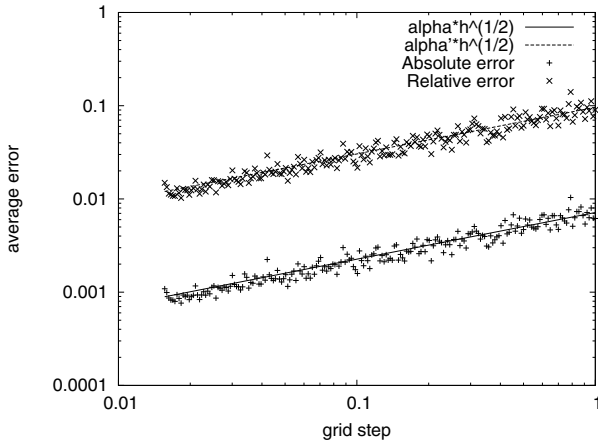
$$\epsilon_{rel}(p) = \frac{\epsilon_{abs}(p)}{\hat{\kappa}(\partial E, p')} \quad (7)$$

The average absolute and relative error have been plotted against the grid step h in Fig. 3b. We experimentally observed that the convergence speed is $\Theta(h^{\frac{1}{2}})$.

Note however that the discrete length of the smallest MDCA grows more likely in $\Theta(h^{-0.43})$ (Fig. 3a), which means that the proposed estimator may not be *uniformly* convergent.



(a)



(b)

Fig. 3. The set of MDCAs has been computed from the contour of the digitization of an ellipse of elongation $\frac{1}{3}$. The minimal, average and maximal discrete length of the MDCAs have been plotted against the grid step h in (a). The average absolute and relative error have been plotted against the grid step h in (b).

The position and the orientation of the ellipses have no significant impact on the discrete length of the MDCAs (and as expected, the finer the resolution is, the smaller the variation is). The elongation of the ellipses has however an impact on the magnitude of the discrete length of the MDCAs. We set the great

axis of an ellipse to 24 pixels and set the small one to 6, 8, 12, 16 and 18 pixels, in order to test ellipses of elongation $\frac{1}{4}$, $\frac{1}{3}$, $\frac{1}{2}$, $\frac{2}{3}$ and $\frac{3}{4}$ respectively. In either case, the average discrete length of the MDCAs grows in $\Theta(h^{-\frac{1}{2}})$ as h tends towards 0. Though, for a given great axis, the larger the small axis is, the higher the average discrete length of the MDCAs is.

5 Comparisons

We report below the comparison of our estimator to the latest curvature estimators of the literature for three shapes: a disk of radius 15, an ellipse of great (resp. small) axis 30 (resp. 20) and a flower of five petals, of great (resp. small) radius 15 (resp. 10).

These shapes have been digitized (according to Gauss scheme) at three different grid steps: $h = 1$, $h = 0.1$ and $h = 0.01$. The digital objects obtained for $h = 1$ and $h = 0.1$ are depicted Fig. 4. Their digital contour is the input of the curvature estimation methods.

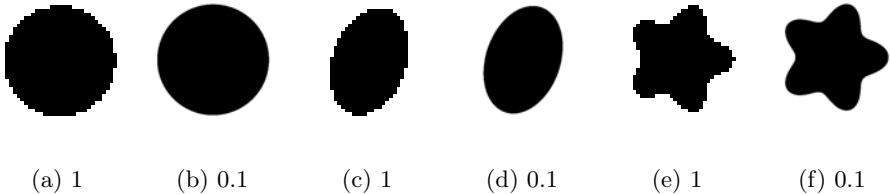


Fig. 4. Digitization at two different grid steps of the circle (a-b), the ellipse (c-d) and the flower (e-f) for which the curvature has been estimated

5.1 Implementation Issues

The GMC estimator [9] computes the shape whose digitization coincides with the input digital object and that minimizes its squared curvature. The minimization is performed by an iterative numerical technique that stops when the difference between the squared curvature of the last two solution shapes is less than a small quantity set to 1.10^{-8} in what follows.

The BC estimator [13,3] is computed from derivative estimations, get by finite differences after the convolution of the contour points by a binomial kernel of a given size. The mask size is an input parameter that is not easy to determine, but following [13], it has been set to $d.h^{\frac{4}{3}}$ where d is the diameter of the shape, equal here to 30 for the circle, the ellipse and the flower.

As explained in Section 2, the proposed estimator is computed from the set of MDCAs.

For the recognition of the DCAs, we use the on-line algorithm proposed in [15]. The algorithm incrementally computes one (Euclidean) circle that separates the

interior and exterior points of a DCA and that is called below *current circle*. Deciding whether a DCA can be extended to a new grid edge requires to check the location of the interior and exterior points of the grid edge with respect to the current circle. If the interior (resp. exterior) point is located inside (resp. outside) the current circle, nothing has to be done because the current circle is still separating. However, if the interior (resp. exterior) point is located outside (resp. inside) the current circle, then either the point is located on another separating circle, or the sets of interior and exterior points are not circularly separable at all. A linear-time procedure, which computes the set of separating circles passing through a given point, is used in the aim of deciding between these two alternatives. A naive upper bound for the extension of a DCA to a new grid edge is thus $O(n)$, but we experimentally observe a constant time in average because the linear-time procedure is called only a few times.

The set of MDCAs is then computed DCA by DCA using the above recognition algorithm and following the scheme given in [4] for the maximal digital straight segments. The mechanism can be coarsely described as follows: given a MDCA A_i and the first grid edge e following A_i , the next MDCA A_j is computed in two steps. First, we compute the longest DCA starting from e and scanning the contour backward. Then, we extend this DCA forward as far as possible until it is maximal. The number of times we try to extend a DCA is bounded by the sum of the discrete length of the MDCAs. We experimentally observe that this sum is proportional to the length of the contour and that the global complexity the MDCAs computation is linear in practice.

5.2 Accuracy and Running Time

In Fig. 5 we compare the curvature plots derived from the MDCA, GMC and BC estimators to the ground-truth. The visual deviation between the estimated graphs and the ground-truth graph reflects the average absolute error available in Tab. 1. For either estimator, the curvature estimations are more accurate for the circle than for the ellipse and more accurate for the ellipse than for the flower. For either shape, the GMC and MDCA estimations gets closer to the ground-truth (Fig. 5) and their absolute error decreases (Tab. 1) as h decreases. However, the BC estimations are not improved by increasing the resolution, except for the flower. Lastly, MDCA estimations are always better than GMC and BC estimations except in one case (maximal error for the flower at grid step 0.01), even by several orders of magnitude for the circle. For the ellipse and the flower, at grid step 0.01, MDCA estimations are 4 times better than GMC estimations in average and much more better than BC estimations (Tab. 1). In Fig. 5 the MDCA graph is hardly confounded with the ground-truth graph.

Tab. 2 reports the running times. The algorithms have been implemented in C++ and have been run on an Intel Core Duo processor work-station with a 2.4 GHz Clock and 2GB of main memory.

The running times of the BC estimator grow quickly as h decreases and is high at grid step 0.01 (18s) because the mask size must be set to $d.h^{4/3}$. The GMC estimator and leads to rather low running times, minimal for the flower (823ms

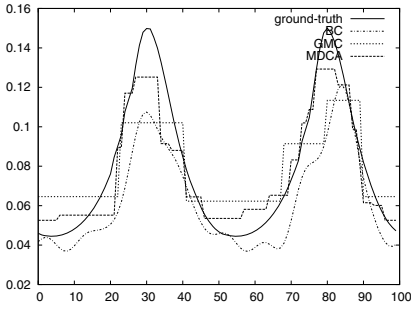
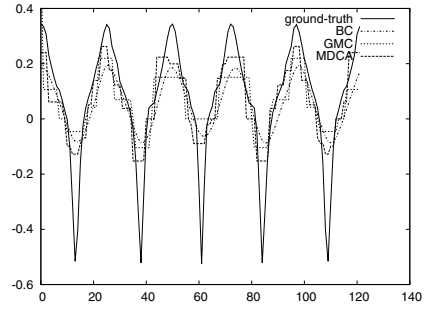
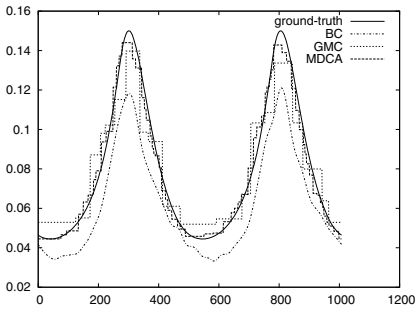
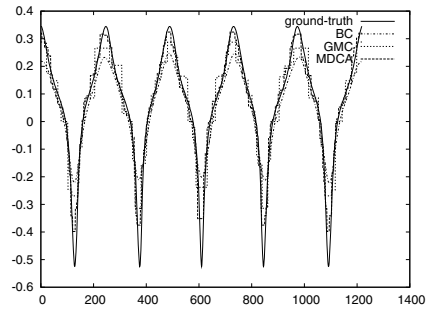
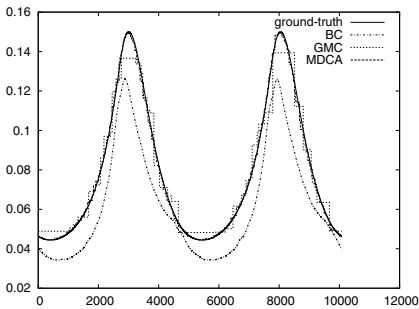
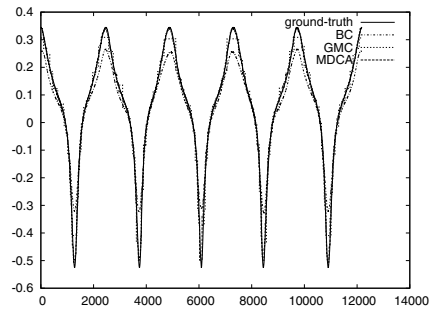
(a) ellipse, $h = 1$ (b) flower, $h = 1$ (c) ellipse, $h = 0.1$ (d) flower, $h = 0.1$ (e) ellipse, $h = 0.01$ (f) flower, $h = 0.01$

Fig. 5. Curvature plots for two shapes digitized at three different resolutions, computed from the proposed estimator (MDCA) and the latest curvature estimators of the literature (BC [13] and GMC [9])

Table 1. Average and maximal absolute error of the curvature estimation for three shapes digitized at three different grid steps and comparisons with the BC and GMC estimators

		circle			ellipse			flower		
h		1	0.1	0.01	1	0.1	0.01	1	0.1	0.01
‡ grid edges		120	1200	12000	100	1008	10078	122	1220	12184
BC	avg	0.0143	0.0143	0.0143	0.0168	0.0170	0.0170	0.1044	0.0623	0.0440
	max	0.0262	0.0205	0.0203	0.0577	0.3634	0.0389	0.4648	0.3236	0.2143
GMC	avg	$8 \cdot 10^{-4}$	$3 \cdot 10^{-5}$	$3 \cdot 10^{-5}$	0.0168	0.0086	0.0044	0.1103	0.0519	0.0222
	max	$5 \cdot 10^{-3}$	$4 \cdot 10^{-5}$	$9 \cdot 10^{-5}$	0.0562	0.0336	0.0281	0.5250	0.2869	0.1529
MDCA	avg	$4 \cdot 10^{-5}$	$5 \cdot 10^{-7}$	$8 \cdot 10^{-8}$	0.0094	0.0034	0.0009	0.0845	0.0281	0.0084
	max	$4 \cdot 10^{-5}$	$5 \cdot 10^{-7}$	$8 \cdot 10^{-8}$	0.0413	0.0149	0.0068	0.4356	0.1918	0.1552

Table 2. Running time of the curvature estimation for three shapes digitized at three different grid steps and comparisons with the BC and GMC estimators

		circle			ellipse			flower		
h		1	0.1	0.01	1	0.1	0.01	1	0.1	0.01
‡ grid edges		120	1200	12000	100	1008	10078	122	1220	12184
timing (ms)	BC	< 1	63	18596	< 1	55	30197	< 1	83	18704
	GMC	< 1	820	2360	< 1	27	1843	< 1	15	843
	MDCA	< 1	23	455	7	139	1783	3	127	2195

at 0.01) and maximal for the circle (2360ms at 0.01). The MDCA estimator is linear-time in practice and also leads to low running times, minimal for the circle (455ms at 0.01) and maximal for the flower (2195ms at 0.01). It is the best method when averaging over the three shapes.

6 Conclusion and Perspectives

The proposed estimator does not require any parameter, is fast to compute (linear-time in practice) and more accurate than the latest estimators of the literature at low as well as high resolution. We provide a necessary condition about the length of the maximal digital circular arcs under which the proposed estimator is multigrid convergent. We experimentally observed that this condition is fulfilled and proving this fact is our main perspective. We are also interested in dealing with shapes corrupted by noise in a multi-resolution framework.

References

1. 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)

2. Coeurjolly, D., Svensson, S.: Estimation of curvature along curves with application to fibres in 3D images of paper. In: Bigun, J., Gustavsson, T. (eds.) SCIA 2003. LNCS, vol. 2749, pp. 247–254. Springer, Heidelberg (2003)
3. 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)
4. Feschet, F., Tougne, L.: Optimal time computation of the tangent of a discrete curve: Application to the curvature. In: Bertrand, G., Couprie, M., Perrotton, L. (eds.) DGCI 1999. LNCS, vol. 1568, pp. 31–40. Springer, Heidelberg (1999)
5. 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)
6. Fleischmann, O., Wietzke, L., Sommer, G.: A Novel Curvature Estimator for Digital Curves and Images. In: Goesele, M., Roth, S., Kuijper, A., Schiele, B., Schindler, K. (eds.) DAGM 2010. LNCS, vol. 6376, pp. 442–451. Springer, Heidelberg (2010)
7. Gross, A., Latecki, L.: Digitizations preserving topological and differential geometric properties. *Comput. Vis. Image Underst.* 62(3), 370–381 (1995)
8. Hermann, S., Klette, R.: A comparative study on 2d curvature estimators. In: Proc. ICCTA, pp. 584–589 (2007)
9. Kerautret, B., Lachaud, J.-O.: Curvature estimation along noisy digital contours by approximate global optimization. *Pattern Recognition* 42(10), 2265–2278 (2009)
10. Klette, R., Rosenfeld, A.: *Digital Geometry - Geometric Methods for Digital Picture*. Morgan Kaufmann, San Francisco (2004)
11. Lachaud, J.-O., Vialard, A., de Vieilleville, F.: Fast, accurate and convergent tangent estimation on digital contours. *Image and Vis. Comput.* 25(10), 1572–1587 (2007)
12. Liu, H., Latecki, L., Liu, W.: A unified curvature definition for regular, polygonal, and digital planar curves. *Int. Jour. of Comput. Vis.* 80, 104–124 (2008)
13. 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)
14. Marji, M.: On the detection of dominant points on digital planar curves. PhD thesis, Wayne State University, Detroit, Michigan (2003)
15. Roussillon, T., Sivignon, I., Tougne, L.: On three constrained versions of the digital circular arc recognition problem. In: Brlek, S., Reutenauer, C., Provençal, X. (eds.) DGCI 2009. LNCS, vol. 5810, pp. 34–45. Springer, Heidelberg (2009)
16. Vialard, A.: Geometrical parameters extraction from discrete paths. In: Miguet, S., Ubéda, S., Montanvert, A. (eds.) DGCI 1996. LNCS, vol. 1176, pp. 24–35. Springer, Heidelberg (1996)
17. Worring, M., Smeulders, A.W.M.: Digital curvature estimation. *CVGIP: Image Understanding* 58(3), 366–382 (1993)

Combining Topological Maps, Multi-Label Simple Points, and Minimum-Length Polygons for Efficient Digital Partition Model

Guillaume Damiand¹, Alexandre Dupas², and Jacques-Olivier Lachaud³

¹ Université de Lyon, CNRS, LIRIS, UMR5205, 69622, France
guillaume.damiand@liris.cnrs.fr

² Université de Poitiers, CNRS, SIC-XLIM, UMR6172, 86962, France
Inserm, Unit 698, 75018 Paris, France
alexandre.dupas@gmail.com

³ Université de Savoie, CNRS, LAMA, UMR5127, 73376, France
jacques-olivier.lachaud@univ-savoie.fr

Abstract. Deformable models have shown great potential for image segmentation. They include discrete models whose combinatorial formulation leads to efficient and sometimes optimal minimization algorithms. In this paper, we propose a new discrete framework to deform any partition while preserving its topology. We show how to combine the use of multi-label simple points, topological maps and minimum-length polygons in order to implement an efficient digital deformable partition model. Our experimental results illustrate the potential of our framework for segmenting images, since it allows the mixing of region-based, contour-based and regularization energies, while keeping the overall image structure.

Keywords: Topological Map, ML-Simple Point, Minimum-Length Polygon, Deformable Model, Interpixel Boundaries, Multi-Label Image.

1 Introduction

Energy-minimizing techniques are now widely spread approaches for segmenting images into meaningful regions. They express the problem as a balance between several energies: (i) *fit-to-data*: energies that express the consistency of the regions with the data, (ii) *regularization*: energies that enforce spatial coherence. Fit-to-data is either region-based (squared error for homogeneity) or contour-based (strong gradient). Regularization is generally enforced through boundary length penalization, sometimes curvature penalization or some a priori knowledge on the probability distribution of the partition model.

Energy minimization can be solved in the continuous world with iterative solving of partial differential equations, leading to snakes and geometric or geodesic active contours [13,34] and Mumford-Shah approximation [19,25,20]. Energy minimization can also be expressed in the discrete world, generally as a graph-cut problem [15] or a Markov Random Field (e.g., see [24]). To sum up, continuous approaches achieve optimality for specific energies. Discrete approaches

are close to the optimal for a wider set of energies, to the price of less pertinent regularization energies: for instance, the length of the boundary is approximated as its staircase length.

As noted by Meltzer *et al.* [17] or Szeliski *et al.* [24], appropriate energies are more important than optimality for good segmentation results. In this paper, we therefore focus on defining an energy-minimizing framework for image segmentation that is as versatile as possible to define energies and relations between regions. We propose a purely discrete framework, which encodes both the topology and the geometry of an image partition. This digital partition is “deformable” in the sense that pixels may change their label during the minimization. The deformation is controlled so that the partition may not change topology during the deformation. This property is desirable in many image analysis applications, like segmentation with a priori knowledge on the output topology (as often in biomedical image segmentation) or atlas matching. This aspect has even been recognized by the level-set community: although the level-set principle allows arbitrary topology changes, several authors have proposed methods to reintroduce topology control in the segmentation (e.g., see [22]). This paper extends the work of [9] because our framework incorporates a new length penalizer with better regularization properties. Moreover, it combines in a single balance the energies: on one side, length penalization and alignments of boundaries with strong gradients define one energy term (in the same spirit as geodesic active contours [4]), on the other side region homogeneity.

Our objective is to define a digital deformable partition model. To that effect, several bottlenecks must be solved. First, we need to describe digital contours as closed paths, eventually intersecting themselves. We use *interpixel* topology [14] which allows to have in digital spaces good topological properties similar to the ones in continuous spaces. Second, we need to control the topology of the deformable partition. This is achieved thanks to *multi-label simple points* [11,9]. Last, during deformations, we need a data structure to manage various queries like access to image statistics in region, estimation of the length of boundaries, etc. We use *topological maps* [8] to describe all the cells composing the deformable partition, and all the incidence and adjacency relations between these cells. Then, we use the *minimum length polygon* [18,23,21] criteria in order to estimate the length or evolving regions.

The paper is organized as follows. We recall the main notions of these papers in Section 2 (for further details, interested readers must refer to the original works). Then, we present the deformable process and the energies in Section 3. Illustrative experiments are given in Section 4. Future works are discussed in Section 5.

2 Preliminaries Notions

2.1 Pixels, Image and Regions

A pixel is an element of the discrete space \mathbb{N}^2 . Two pixels $p = (x, y)$ and $p' = (x', y')$ are *4-adjacent* if $|x - x'| + |y - y'| = 1$. A *4-path* between two pixels

p and p' is a sequence of pixels ($p = p_1, \dots, p_k = p'$) such that each couple of consecutive pixels are 4-adjacent. A set of pixels S is *4-connected* if there is a 4-path between any couple of pixels in S using all its pixels in S .

An image is a set of pixels. In this work, we consider *labeled images*, *i.e.* images in which each pixel is associated with a label (any value). In labeled images, a *region* is a maximal set of 4-connected pixels. Thus, regions form a partition of the image: two different regions are disjoint, and the union of all the regions is equal to the image. Moreover, we consider a specific region R_0 , called the *infinite region*, which is the complementary of the image. Two regions are said *4-adjacent* if there is a pixel in the first region that is 4-adjacent to a pixel in the second region.

2.2 Interpixel Topology and Cubical Complexes

In interpixel topology, we consider the cellular decomposition of the euclidean space \mathbb{R}^2 into regular grids. *Pixels* are elements of dimension 2 (squares) of the decomposition, *linels* are elements of dimension 1 (segments) between pixels, and *pointels* are elements of dimension 0 (points) between linels. We call *i-face* an element of dimension i (see examples in Fig. 1).

For a pixel p , we note $\text{linels}(p)$ the set of the four linels between p and its four 4-neighbors, and $\text{pointels}(p)$ the set of the four pointels around p . For a linel l , we note $\text{pointels}(l)$ the set of the two pointels around l . These notations are directly extended to set of elements. For example, we note $\text{pointels}(L)$, L being a set of linels, the union of $\text{pointels}(l)$ for all $l \in L$.

A *cubical complex* C is a set of pointels, linels, surfels, glued together by adjacency and incidence relations. A *free pair* in C is a couple (c, c') with $c \in C$ being an i -face ($0 \leq i \leq 1$), $c' \in C$ being an $(i + 1)$ -face incident to c , and such that there is no other $(i + 1)$ -face in C distinct from c' incident to c . In such a

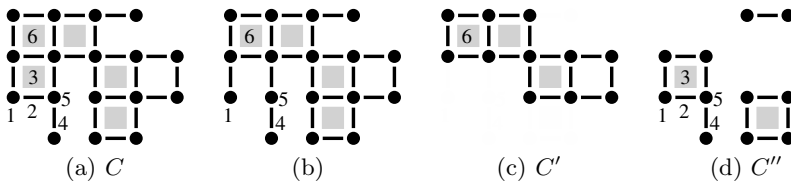


Fig. 1. Example of interpixel topology, cubical complex and collapse. (a) A cubical complex C . Pointels are black disks (for example 1 and 5), linels are black segments (for example 2 and 4) and pixels are gray squares (for example 3 and 6). Pointel 1 is incident to linel 2 and to pixel 3. Linel 2 is incident to pixel 3. Pointels 1 and 5 are adjacent. Linels 2 and 4 are adjacent. Couple $(2, 3)$ is a free pair. (b) Cubical complex obtained from C by the elementary collapse of free pair $(2, 3)$. (c) A cubical complex C' such that C collapses onto C' . (d) A cubical complex C'' such that C'' is simple for C' . “Removing” C'' from C gives C' . Symmetrically, C'' is add-simple for C' and “adding” C'' in C' gives C .

case, (c, c') can be removed from C without modifying its topology. This is the basic operation used to deform a cubical complex while preserving its topology called the *elementary collapse*.

A cubical complex C collapses onto a second cubical complex C' if there is a sequence of *elementary collapse* transforming C into C' . We say that a complex Y is *simple* for X if Y can be “removed” from X without modifying the topology of X , and symmetrically, a complex Y' is *add-simple* for X' if Y' can be “added” into X' without modifying the topology of X' . These two notions can be defined thanks to the collapse operation (see [6] for precise definitions, and some examples in Fig. [1]).

2.3 Multi-label Simple Points

Given a pixel x belonging to region X , and a region R , we note $l(x, R)$ the linels around x that “touch” R : $l(x, R) = \{\text{linels } l | l \in \text{linels}(x) \text{ and the second pixel around } l \text{ belongs to } R\}$. Similarly we note $p(x, R) = \{\text{pointels } p | p \in \text{pointels}(x) \text{ and the three pixels around } p \text{ distinct from } x \text{ belong to } R\}$. The *contour* of x touching R , called $c(x, R)$ is the composition of these two sets: $c(x, R) = (p(x, R), l(x, R))$. We note $c^-(x, R) = (p^-(x, R), l(x, R))$, with $p^-(x, R)$ the set of pointels touching a linel in $l(x, R)$, *i.e.* $p^-(x, R) = \text{pointels}(l(x, R))$.

The *frontier* between two regions X and R is noted $f(X, R)$. This is the set of all the linels between one pixel in X and one pixel in R , denoted $l(X, R)$, plus all the pointels touching these linels $p(X, R) = \text{pointels}(l(X, R))$.

Definition [1] of multi-label simple points is based on the interpixel topology and the simple and add-simple notions (see examples in Fig. [2]). A pixel x belonging to region X is ML-simple for region R if the flip of x in R does not modify the topology of region R (first condition of the definition), does not modify the topology of region X (second condition of the definition), and if it does not modify the different frontiers around x (last condition of the definition). For the two first conditions, a cubical complex C is said homotopic to a 1-disk if it can be collapsed onto a complex that is composed only by one linel. For the last condition, intuitively it means that two regions adjacent along one frontier before the flip are still adjacent after the flip (*i.e.* the frontier still exists) and that two regions not adjacent before the flip are still not adjacent after.

Definition 1 (ML-simple points). *A pixel x , belonging to region X , is ML-simple for region R if:*

1. $c(x, R)$ is homotopic to a 1-disk;
2. $c(x, X)$ is homotopic to a 1-disk;
3. for each region O 4-adjacent to x , distinct from X and R : $c^-(x, O)$ is simple for $f(X, O)$; and add-simple for $f(R, O)$.

The main operation used during the deformation algorithm is the *flip* of a pixel $x \in X$ into a given region R . This operation consists in removing x from its initial region X and adding it to region R . It has been proved in [11] that if the pixel x is ML-simple point for region R , it can be flipped into region R without

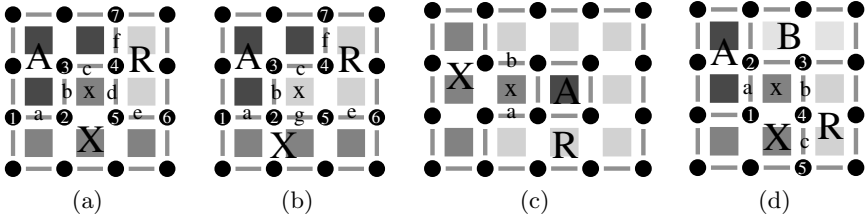


Fig. 2. Examples of ML-simple points and non ML-simple points. (a) x is ML-simple for R . $c(x, R) = (\{ \}, \{d\})$ and $c(x, X) = (\{ \}, \{g\})$ are homotopic to 1-disks. $c^-(x, A) = (\{2, 3, 4\}, \{b, c\})$ is simple for $f(X, A) = (\{1, 2, 3, 4\}, \{a, b, c\})$; and add-simple for $f(R, A) = (\{4, 7\}, \{f\})$. (b) Partition obtained from (a) after the flip of x in R . $f(X, R) = (\{2, 5, 6\}, \{e, g\})$, $f(X, A) = (\{1, 2\}, \{a\})$, $f(R, A) = (\{2, 3, 4, 7\}, \{b, c, f\})$. (c) x is not a ML-simple point for R because $c(x, R) = (\{ \}, \{a, b\})$ is not homotopic to a 1-disk. Flipping x in R modifies the topology of R . (d) x is not a ML-simple point for R because $c^-(x, A) = (\{1, 2\}, \{a\})$ is not add-simple for $f(R, A) = (\{ \}, \{ \})$. Flipping x in R modifies the adjacency between regions.

modifying the topology of the partition. This operation modifies regions R and X , but also some frontiers. Since pointels are deduced from linels by definition of frontiers, modifications only concern the linels of the frontiers (see example in Fig. 2):

- $l(X, R) \leftarrow l(X, R) \setminus l(x, R) \cup l(x, X)$;
- For any region O with $O \neq X, O \neq R$: $l(X, O) \leftarrow l(X, O) \setminus l(x, O)$;
- For any region O with $O \neq X, O \neq R$: $l(R, O) \leftarrow l(R, O) \cup l(x, O)$;

2.4 Minimum Length Polygon

The Minimum Length Polygon (MLP) is a classical polygonalization method of digital contours [18,23]. Assuming that the digital contour is the boundary of some digital object, it is the polygon with shortest perimeter, whose interior contains the centers of the pixels of the object and whose exterior contains the centers of the pixels of the background. One may also see it at the shortest closed polygonal line which stays within the strip formed by the minkowski sum of the digital contour and a centered unit square while making one loop inside it (see Fig. 3). It has been proved recently that the MLP is a good regularizer for digital active contours [10]: for instance, it is a kind of convex energy in a digital space. We use the combinatorial MLP in our experiments, since it is one of the fastest method for computing it [21].

2.5 Topological Map

Several works have proposed data structures to describe image partitions [12,2]. The common goal is to represent regions of a given partition and all the incidence

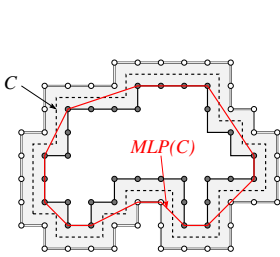


Fig. 3. Minimum Length Polygon of the digital contour C

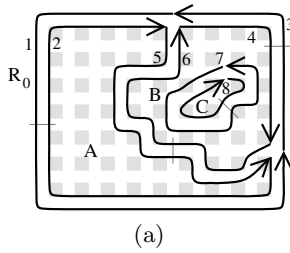
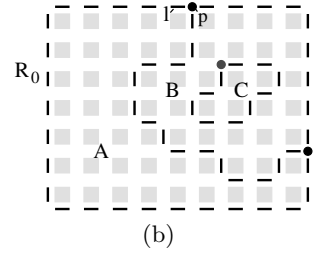


Fig. 4. Example of topological map. (a) Combinatorial map describing the topology of frontiers. (b) Interpixel matrix describing the geometry of frontiers.



and adjacency relations between regions. Structures based on *combinatorial maps* [15] have several advantages justifying their use: they represent all the cells of the subdivision (vertices, edges and faces) and all the incidence and adjacency relations between these cells; they are defined based on a unique element called *dart*, simplifying their use; and they are defined in any dimension which allows extensions of this work in higher dimension.

Topological maps are an extension of combinatorial maps created to describe the whole topology of image partitions [8]. A topological map is composed of a combinatorial map that describes the cells of the partition and the incidence and adjacency relations between these cells (e.g. Fig. 4(a)), plus an interpixel matrix that describes the geometry of regions (e.g. Fig. 4(b)).

The combinatorial map describes each adjacency relation between two regions by an edge composed of two darts linked together. Thus, a dart can be seen as an half-edge (for example darts 1 and 2 in Fig. 4(a)). A cycle of successive darts describes a contour of a given region (for example (1, 3) or (2, 5)). In the interpixel matrix, each line belonging to one frontier between two regions is switched *on*, and each point touching more than two lines is switched *on*. Other elements are switched *off*. Each dart knows its region, and each region knows one dart of its external contour. Moreover, each dart d is associated with a pair (*pointel*, *linel*) such that *pointel* corresponds to the origin of dart d , and *linel* is the first line of the frontier associated with d (for example dart 1 is associated with pair (p, l)).

3 Deformable Model Process

To propose a new digital deformable partition model, we start by presenting the energies used and their computation algorithms. Then, we detail the main operations which are the test if a pixel is ML-simple, and the flip of a pixel. We study how to update the energies during the flip, trying to keep modifications as local as possible. Last, we give the global algorithm regrouping all these tools to minimize the energy of the digital deformable partition model.

3.1 Energies

We define the global energy of the partition as the weighted sum of two energies: E_d for the energy attached to data, and E_p for the energy attached to the deformable partition. In this work, we use the two following energies: $\mathit{contour}(C)$ is the MLP length of contour C weighted by the pixel gradient; $\mathit{mse}(R)$ is the minimum square error of region R .

These two energies can be directly computed from the topological map. For $\mathit{contour}(C)$, we run through all the darts of the given contour, and for each dart we run through all its linels, providing the geometry of the contour. The corresponding MLP can be computed from this geometry in time linear in the number of linels [21]. Then, we project each linel of the contour on the corresponding edge of the MLP. We compute the length of this projection, and multiply it by one minus the absolute difference between the two pixels around the current linel. This difference is divided by the maximal difference of the image, giving a number between 0 and 1. The value of $\mathit{contour}(C)$ is simply the sum of these values for each linel of the contour. If the image is uniform, $\mathit{contour}(C)$ is equal to the length of the MLP. Otherwise, $\mathit{contour}(C)$ becomes smaller when it separates more contrasted pixels.

For $\mathit{mse}(R)$, we run through each pixel of the image that belongs to the given region, and compute the number of pixels $M_0(R)$, the sum of the pixel values $M_1(R)$ and the sum of the pixel squared values $M_2(R)$. With these three values, we can directly estimate $\mathit{mse}(R) = \frac{1}{M_0(R)}(M_2(R) - \frac{M_1(R)^2}{M_0(R)})$.

To avoid several re-computation of these values, we store them in additional elements associated with the topological map. The value of $\mathit{contour}(C)$ is stored for each contour, and $M_0(R)$, $M_1(R)$ and $M_2(R)$ are stored for each region.

The two energies E_d and E_p are computed by summing up the basic energies on each element of the topological map: $E_d = w_R \times \sum_{\text{region } R} \mathit{mse}(R)$ and $E_p = w_C \times \sum_{\text{contour } C} \mathit{contour}(c)$. The weights w_R , w_C associated to both energies allow us to change the balance between the two energies.

3.2 Operations

The two main operations of the deformable process are the test if a pixel is ML-simple, and the flip of a pixel. Algorithm 1 allows to test if a given pixel is ML-simple. This algorithm follows directly Definition 1 (see [9] for more details).

The two first lines of Algo. 1 correspond to the two first tests of Definition 1. Function `ISDISK`, not given here, is simple to write since $c(x, R)$ is restricted to a small number of cases. Indeed, $c(x, R)$ can be empty, a cycle (i.e. a 2-disk), or made of several connected components: in these cases it is not a disk. The only other possible case is the case of the disk.

The “foreach” loop of Algo. 1 corresponds to the last condition of Definition 1; we only detail the two conditions `add-simple` and `simple` by using the definitions given in [6]. The main principle of these definitions is to test if the given set can be collapsed onto a specific set. This is the role of function `COLLAPSE` not given here. In our case, the first set is $c(x, O)$ which is bounded by the linels and

Algorithm 1. `isSimple(x,R)`

Data: pixel $x \in X$; region R .
Result: *true* iff x is an ML-simple point for R .
if *not* `ISDISK`($c(x, R)$) **then return** *false*;
if *not* `ISDISK`($c(x, X)$) **then return** *false*;
foreach region O 4-adjacent to x , $O \neq X$, $O \neq R$ **do**
 $P_1 \leftarrow \{l \in p(x, O) \mid l \in \text{pointels}(l(X, O) \setminus l(x, O))\}$;
 if *not* `COLLAPSE`($c(x, O), P_1$) **then return** *false*;
 $P_2 \leftarrow \{l \in p(x, O) \mid l \in f(R, O)\}$;
 if *not* `COLLAPSE`($c(x, O), P_2$) **then return** *false*;
return *true*;

pointels incident to the given pixel x . Thus, the collapse function is restricted to a small number of cases which can easily be tested.

The complexity of Algo. 1 is $O(1)$ since each test is restricted to pointels and linels around the considered pixel, and these numbers are both bounded by 4.

Algorithm 2. `flip(x,R)`

Data: a pixel $x \in X$ ML-simple for R .
Result: flip x into region R .
switch off linels in $l(x, R)$;
switch on linels in $l(x, X)$;
foreach dart d touching x **do**
 if $\text{pointel}(d) \in c^-(x, R)$ **then** update pair(d);

The second main operation is the flip which consists in removing the given pixel x from its original region X , and add it into its new region R . To be valid, this operation requires that x is a ML-simple pixel for region R . Algorithm 2 is made of three steps. First we switch off the linels in $l(x, R)$. Second we switch on the linels in $l(x, X)$. Since x is flip in region R , the linels in $l(x, R)$ become “inner linels”, *i.e.* inside region R , while linels in $l(x, X)$ become frontier linels between region R and X . The last step consists to update associations between darts and pairs. This updating is required when the pointel associated with a dart belongs to $c^-(x, R)$ (as in the case shown in Fig. 2(a)). In this case, the flip of x in R involves modifications of the extremities of some frontiers (in Fig. 2(a) $f(X, A) = (\{1, 2, 3, 4\}, \{a, b, c\})$ before the flip, and $f(X, A) = (\{1, 2\}, \{a\})$ after: one extremity has moved).

The complexity of Algo. 2 is also in $O(1)$. Indeed the number of linels in $l(x, R)$ and in $l(x, X)$ is at most 4, and the number of darts touching pixel x is at most 12 (3 darts for each pointel around x).

3.3 Update Energies for Flip

The main operation performed during the energy minimization process consists in flipping a pixel into a given region, and then to update the different energies accordingly. A first solution is to re-compute all the energies as explained in

Sect. 3.1. But this involves a considerable computation time since the flip operation is called many times. A better solution consists in updating incrementally the energies.

For *mse*, we know that only region X and region R are modified. Thanks to the moments stored in each region, we can directly (*i.e.* in $O(1)$) update M_0 , M_1 and M_2 for these two regions and thus the two *mse*. This modification can be made in constant time without additional cost for the `flip` algorithm. However, the problem is more complex for *contour* since there is no algorithm yet able to update a MLP after the modification of one of its pixels. Thus, *contour* is re-computed for the two contours around the flipped pixel by using the same algorithm than for the initialization step. The improvement of this particular point is one of our future goals.

The global energy of the partition can thus be updated by subtracting original values of each energy which will be modified, and by adding the updated energy values after the flip.

3.4 Energy Minimization Algorithm

We present in Algo. 3 the global minimization algorithm taking a topological map as input, and minimizing its energy by flipping ML-simple points. The algorithm starts by an initialization step, first to compute all the energies by using the principles given in Sect. 3.1. Then, each region X is considered to find a possible flip. This is the role of the `possibleFlip(X)` function, which is as follows. The function runs through each line l describing a contour of X . Line l separates two pixels: x in X , and a second pixel in another region R . If x is ML-simple for R , then flipping x into R is a possible candidate. Then, we flip x into R and compute the difference δ between the old energy (before the flip) and the new one (after the flip). Finally, we apply the reverse flip to retrieve the initial configuration and proceed to the next line. If δ is negative, the flip of x into R decreases the global energy. In this case, this flip (x, R) is stored in region X into a variable called `candidate(X)`, and function `possibleFlip` returns true. Otherwise, we continue to run over the lines of the contours of X . If no flip decreasing the energy is found, `possibleFlip` returns false.

At the end of the initialization loop, S contains all regions having a flip that decreases the energy. In the main loop of the algorithm, we choose a region in the set, and flip the corresponding pixel. This operation changes the geometry and the energy values of the regions X and R . Stored flips for X , R , and their adjacent regions can be invalid (for example the considered pixel is not a simple point anymore, or the flip increases the energy) and should be updated using the `possibleFlip` algorithm. We can then proceed with an other region of S .

The main loop finishes when the set of candidate becomes empty. In this case, the minimization process is terminated: there is no possible flip decreasing the global energy.

Algorithm 3. Digital partition minimization

Data: a topological map M ; an image I .
Result: minimize the energy of the digital partition.
initialize all the energies of M ; $S \leftarrow \emptyset$;
foreach region X **do**
 | **if** *possibleFlip*(X) **then** $S \leftarrow S \cup \{X\}$;
while S is not empty **do**
 | $X \leftarrow$ a region in S ; $S \leftarrow S \setminus X$;
 | $(x, R) \leftarrow$ *candidate*(X);
 | *flip*(x, R);
 | **if** *possibleFlip*(X) **then** $S \leftarrow S \cup \{X\}$;
 | **foreach** region O 4-adjacent to X **do**
 | | $S \leftarrow S \setminus O$;
 | | **if** *possibleFlip*(O) **then** $S \leftarrow S \cup \{O\}$;

4 Experiments

In this section, we first give some deformation results on artificial image using various weights w_R and w_C . Then, we show how the digital deformable partition model is used to fit an initial partition with two pictures from the Berkeley Segmentation Dataset [16]. In the following, weight values w_R and w_C are written as the ratio w_R/w_C .

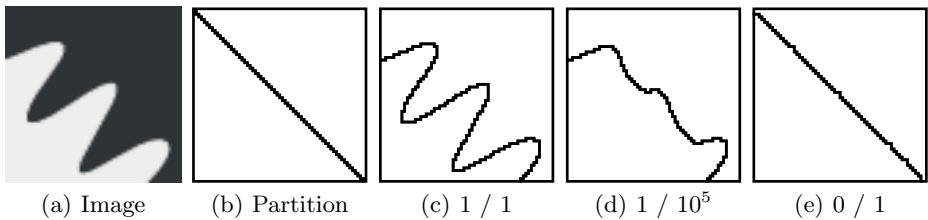


Fig. 5. Experimentation on an artificial image with two simple regions

In Fig. 5, we deform the initial partition (Fig. 5(b)) of an image containing two regions (Fig. 5(a)). Different weights are used to illustrate how their values modify the resulting partition. If the weight of the contour energy is low ($w_R/w_C < 1/10^3$ as for example in Fig. 5(c)), the value of the region energy is greater than the value of the contour energy: the contour energy does not change the resulting partition. As the weight of the contour energy increases, the curve separating the two regions becomes straighter (Fig. 5(d)). If the weight of the region energy is set to zero, the obtained contour is nearly a straight line (Fig. 5(e)). It does not exactly match the frontiers of the initial partition: where the border crosses the curve, the gradient part of the contour energy slightly alter the shape of the border and produces some irregularities.

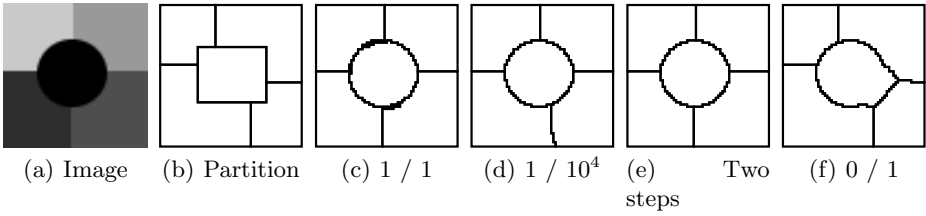


Fig. 6. Deformation of an artificial image with five regions

In Fig. 6 we use the deformation process on an artificial image with five regions (Fig. 6(a)). The frontiers of the initial partition, presented in Fig. 6(b), are not aligned with the border of the different shape in the original image. The deformation process is applied with two different ratio between the energy weights. When the ratio between w_R and w_C is high (Fig. 6(c)), the region energy is privileged and borders follow correctly the image data. But we see that there is some artifact around the disk region: they are caused by the starting point of the different edges which do not move during the deformation process. When the ratio is lower, for instance $1/10^4$, some of the borders move correctly. However the edge separating the two darker gray region is not positioned correctly (Fig. 6(d)): the energy cost of having a curved border outweighs the gain in the region mse . Thus, the border is not able to move toward the separation of the two regions. To solve this issue, we applied a two step deformation with first a $1/1$ ratio and then a $1/10^4$ ratio: the produced segmentation fits correctly the image data without artifact (Fig. 6(e)). Figure 6(f) presents the partition obtained using the contour energy alone: some parts of the disk are retrieved thanks to the gradient measure used in the energy. The bottom right side of the disk is straighten to minimize the length of the contour. If we use a 2D energy based on the number of linels, similar to the one used in 9, the initial partition has already a minimal contour energy. This shows the improvement of the regularization power of the new contour energy based on the MLP and the gradient.

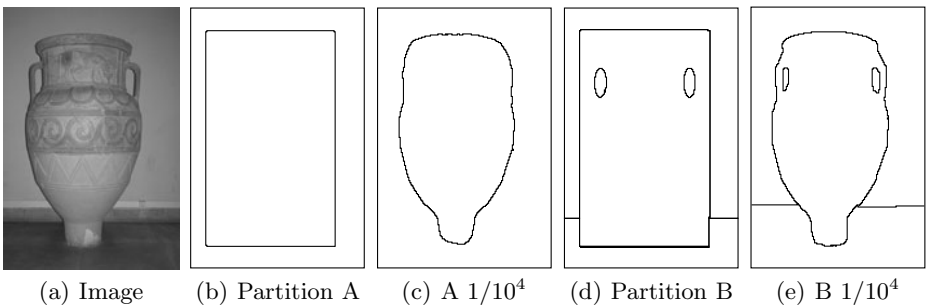


Fig. 7. Deformation of different initial partitions of a picture

The first real image is presented Fig. 7. Using the partition presented in Fig. 7(b), the deformation produced the partition shown in Fig. 7(c) with a $1/10^4$ weight ratio. We note that the shape of the object approximate the amphora in the picture but the contours are not perfectly matched. Using the partition presented in Fig. 7(d), the result (Fig. 7(e)) also fits correctly the amphora but for instance the curved shape just up from the handles is not correctly matched due to a high MLP value.

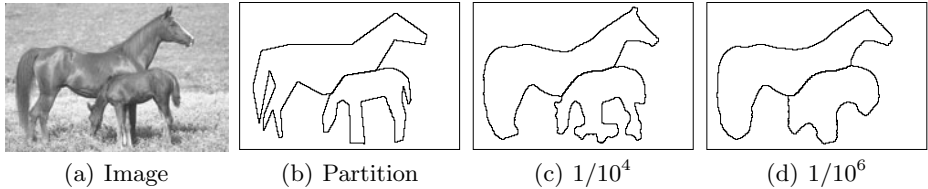


Fig. 8. Deformation of an initial partition of a picture with different weight ratio

In Fig. 8, we show deformation processes applied on a picture containing two horses (Fig. 8(a)). With the initial partition, Fig. 8(b), the first deformation guided by a weight ratio of $1/10^4$ fit the horses shapes (Fig. 8(c)): the legs of the small horse are segmented with the dark area of grass around. This is caused by the region energy which decreases when dark area are grouped. In Fig. 8(d), the weight ratio is $1/10^6$: the smaller horse is poorly segmented and the contour of the taller horse are approximated. We show that the contours tend to be as straight as possible: this remove thin parts of the regions.

5 Conclusion

In this paper, we have presented a new method of deformable digital partition. Several works have shown the interest of using deformable models for image segmentation, but to our knowledge, no work has yet be done on deforming any partition while preserving its topology. To achieve this objective, we used: (1) topological maps to efficiently compute and store features associated with the image and the partition; (2) minimum length polygons as estimator tools; and (3) ML-simple points to guarantee the preservation of the topology.

Our experiments show the interest of this approach to deform an initial partition according to image data. We can, for instance, propose a guided segmentation tool where a user creates an initial partition that is fitted to image data using the deformation process. We can also foresee the use of the deformable partition to improve the result of segmentation algorithms according to our energies.

In future works, we will focus on improving the update of the MLP after a flip operation. The MLP is currently completely re-computed after flips, involving important computation time. This can be improved by defining dynamic MLP capable of incremental update. Another goal is to use other energies to

improve our results. We can, for example, use edge-detector filters (as Canny or Marr-Hildreth detectors) instead of a simple gradient energy, or add geometrical energies on regions to take their shapes into account (for example stretching or compaction). Finally, we could consider the extension of this work in 3D. Since topological maps and ML-simple points have already been defined in 3D [7,9], we need to study the extension of MLP in higher dimension.

References

1. Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell.* 23(11), 1222–1239 (2001)
2. Brun, L., Domenger, J.-P., Braquelaire, J.-P.: Discrete maps: a framework for region segmentation algorithms. In: *Proc. GBR*, pp. 83–92 (1997)
3. Caselles, V., Catte, F., Coll, T., Dibos, F.: A geometric model for active contours. *Numerische Mathematik* 66, 1–31 (1993)
4. Caselles, V., Kimmel, R., Sapiro, G.: Geodesic active contours. *Int. J. Comput. Vision* 22(1), 61–79 (1997)
5. Couprie, C., Grady, L., Najman, L., Talbot, H.: Power watersheds: A new image segmentation framework extending graph cuts, random walker and optimal spanning forest. In: *Proc. ICCV*, pp. 731–738. IEEE, Los Alamitos (2009)
6. Couprie, M., Bertrand, G.: New characterizations of simple points, minimal non-simple sets and P-simple points in 2D, 3D and 4D discrete spaces. In: Coeurjolly, D., Sivignon, I., Tougne, L., Dupont, F. (eds.) *DGCI 2008*. LNCS, vol. 4992, pp. 105–116. Springer, Heidelberg (2008)
7. Damiand, G.: Topological model for 3d image representation: Definition and incremental extraction algorithm. *Computer Vision and Image Understanding* 109(3), 260–289 (2008)
8. Damiand, G., Bertrand, Y., Fiorio, C.: Topological model for two-dimensional image representation: Definition and optimal extraction algorithm. *Computer Vision and Image Understanding* 93(2), 111–154 (2004)
9. Damiand, G., Dupas, A., Lachaud, J.-O.: Fully deformable 3d digital partition model with topological control. *Pattern Recognition Letters* (to appear 2011)
10. de Vieilleville, F., Lachaud, J.-O.: Digital Deformable Model Simulating Active Contours. In: Brlek, S., Reutenauer, C., Provençal, X. (eds.) *DGCI 2009*. LNCS, vol. 5810, pp. 203–216. Springer, Heidelberg (2009)
11. Dupas, A., Damiand, G., Lachaud, J.-O.: Multi-label simple points definition for 3D images digital deformable model. In: Brlek, S., Reutenauer, C., Provençal, X. (eds.) *DGCI 2009*. LNCS, vol. 5810, pp. 156–167. Springer, Heidelberg (2009)
12. Fiorio, C.: A topologically consistent representation for image analysis: the frontiers topological graph. In: Miguët, S., Ubéda, S., Montanvert, A. (eds.) *DGCI 1996*. LNCS, vol. 1176, pp. 151–162. Springer, Heidelberg (1996)
13. Kass, M., Witkin, A., Terzopoulos, D.: Snakes: Active contour models. *Int. J. Comput. Vision* 1(4), 321–331 (1988)
14. Khalimsky, E., Kopperman, R., Meyer, P.R.: Computer graphics and connected topologies on finite ordered sets. *Topology and its Applications* 36, 1–17 (1990)
15. Lienhardt, P.: N-dimensional generalized combinatorial maps and cellular quasi-manifolds. *Int. J. Comput. Geom. Ap.* 4(3), 275–324 (1994)
16. Martin, D., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: *Proc. ICCV*, vol. 2, pp. 416–423 (2001)

17. Meltzer, T., Yanover, C., Weiss, Y.: Globally optimal solutions for energy minimization in stereo vision using reweighted belief propagation. In: Proc. ICCV, vol. 1, pp. 428–435 (2005)
18. Montanari, U.: A note on minimal length polygonal approximation to a digitized contour. *Communications of the ACM* 13(1), 41–47 (1970)
19. Mumford, D., Shah, J.: Optimal approximations by piecewise smooth functions and associated variational problems. *Comm. Pure Appl. Math.* 42, 577–684 (1989)
20. Pock, T., Chambolle, A., Cremers, D., Bischof, H.: A convex relaxation approach for computing minimal partitions. In: Proc. CVPR, pp. 810–817 (2009)
21. Provençal, X., Lachaud, J.-O.: Two linear-time algorithms for computing the minimum length polygon of a digital contour. In: Brlek, S., Reutenauer, C., Provençal, X. (eds.) DGCI 2009. LNCS, vol. 5810, pp. 104–117. Springer, Heidelberg (2009)
22. Ségonne, F.: Active contours under topology control - genus preserving level sets. *Int. Journal of Computer Vision* 79, 107–117 (2008)
23. Sklansky, J., Chazin, R.L., Hansen, B.J.: Minimum perimeter polygons of digitized silhouettes. *IEEE Trans. Computers* 21(3), 260–268 (1972)
24. Szeliski, R., et al.: A comparative study of energy minimization methods for Markov random fields with smoothness-based priors. *IEEE Trans. Pattern Anal. Mach. Intell.* 30(6), 1068–1080 (2008)
25. Vese, L.A., Chan, T.F.: A multiphase level set framework for image segmentation using the Mumford and Shah model. *Int. J. Comput. Vision* 50(3), 271–293 (2002)

Construction of 3D Orthogonal Cover of a Digital Object

Nilanjana Karmakar¹, Arindam Biswas¹,
Partha Bhowmick², and Bhargab B. Bhattacharya³

¹ Department of Information Technology
Bengal Engineering and Science University, Shibpur, Howrah, India
nilanjana.nk@gmail.com, abiswas@it.becs.ac.in

² Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur, India
bhowmick@gmail.com

³ Advanced Computing and Microelectronics Unit
Indian Statistical Institute, Kolkata, India
bhargab@isical.ac.in

Abstract. The orthogonal cover of a 3D digital object is a minimum-volume 3D polytope having surfaces parallel to the coordinate planes, and containing the entire object so as to capture its approximate shape information. An efficient algorithm for construction of such an orthogonal cover imposed on a background grid is presented in this paper. A combinatorial technique is used to classify the grid faces constituting the polytope while traversing along the surface of the object in a breadth-first manner. The eligible grid faces are stored in a doubly connected edge list, using which the faces are finally merged to derive the isothetic polygons parallel to the coordinate planes, thereby obtaining the orthogonal cover of the object. The complexity of the cover decreases with increasing grid size. The algorithm requires computations in integer domain only and runs in a time linear in the number of voxels constituting the object surface. Experimental results demonstrate the effectiveness of the algorithm.

Keywords: 3D orthogonal cover, DCEL, isothetic polygon, orthogonal polytope, shape analysis.

1 Introduction

Characterization and feature extraction of 3D objects have been an important aspect of 3D image analysis. Several works have been reported for constructing a polyhedron P enclosing a set of integer points S , usually known as the *discrete volume polyhedrization* [6,7]. In general, these algorithms follow the principle of *marching cube* that generates a triangulated polyhedral surface in which local configurations for voxels are modeled by small triangles. However, such an algorithm bears a limitation that the number of triangular faces in the surface happen to be comparable with the number of points in S . Another work on 3D

objects relates to digitizing the shape of an object based on range images [21], which involves combining a collection of range images, captured by the range scanner, to form a polygonal mesh that completely describes the object. A number of such meshes aligned and zippered together forms a continuous surface that correctly captures the topology of the object. Based on the surface representation, different techniques for shape analysis of digitized objects form an important area of research in the 3D domain [8,10,13,19,20].

The marching cubes algorithm triangulates a 3D iso-surface based on cubic cell decomposition, their local configurations, and displacement [7,16,17]. The surface intersects a particular cube at certain edges such that the data value of some of the vertices of the cube is greater than or equal to that of the surface. These vertices lie on or inside the surface while other vertices do not. An edge having two opposite type of vertices, thus classified, is intersected by the surface at locations determined by linear interpolation of the vertices. As a cube consists of eight vertices, each in one of two states (inside or outside the surface), a surface can intersect the cube in at most $2^8 = 256$ possible ways, which are reduced to 14 possible patterns using complementary and rotational symmetries. Such an intersection produces at least one and at most four triangles within the cube. Once the intersections are obtained for all cubes, a *marching* procedure through the subsequent cubes leads to an approximate representation of the iso-surface by a triangular mesh. However, since adjacent cubes in the configuration share edges and vertices, the calculation of edge intersection for each cube can be reduced to three edges instead of twelve edges [16]. Approaches to reduce computational activities have been suggested by several researchers [9,15,23]. Nevertheless, the algorithm can be enhanced to handle multi-resolution rectilinear data [22] and data sets in higher dimensional space [2,3].

This paper presents an efficient algorithm that derives the orthogonal cover of a 3D object in \mathbb{Z}^3 . The problem of construction of inner and outer isothetic/orthogonal covers/frontiers of digital objects in 2D has already been studied in great details [4,5,14]. It is implemented by constructing a doubly connected edge list [1] that maintains complete records of vertices, edges, and faces of the isothetic polygons constituting the cover. Merging the coplanar and contiguous faces of the orthogonal cover begets a compact representation and captures the complete topology without triangulation or any such form of surface decomposition. The algorithm does not consider self-intersecting surfaces or holes lying inside the object, although it can be extended to consider these aspects. Due to the complex topology of a digital object, the 3D orthogonal cover is likely to contain many external concavities, which are gradually revealed as we decrease the grid size, which is a specialty of the proposed algorithm.

The paper is organized as follows. Section 2 contains preliminary definitions and the problem statement. Section 3 presents the proposed algorithm with its step-by-step explanation and time complexity. Experimental results and concluding remarks are given in Section 4.

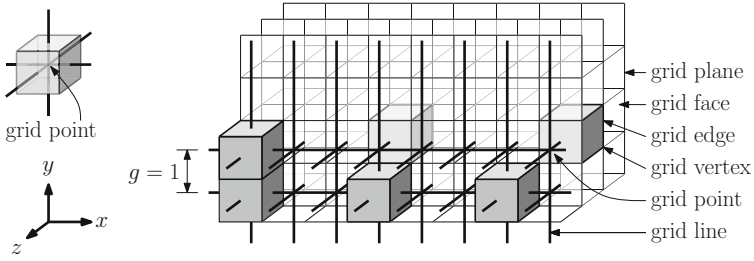


Fig. 1. 3D digital space and 26N [14]. Left: A 3-cell and its corresponding grid point. Right: Three pairs of α -adjacent 3-cells for $\alpha \in \{0, 1, 2\}$, $\alpha \in \{0, 1\}$, and $\alpha = 0$ (from left to right). The 3-cells in each of these three pairs are connected in 26N.

2 Definitions and Preliminaries

Let A be a *3D digital object*, which is defined as a finite subset of \mathbb{Z}^3 , with all its constituent points (i.e., voxels) having integer coordinates. Each voxel is equivalent to a *3-cell* centered at the concerned integer point. As shown in Fig. 1, two 3-cells can be α -adjacent for $\alpha = 0, 1, 2$. Two 3-cells with centers at $(x_1, y_1, z_1) \in \mathbb{Z}^3$ and $(x_2, y_2, z_2) \in \mathbb{Z}^3$ are said to be connected in *26-neighborhood* (26N) if they are $\alpha (\geq 0)$ -adjacent, i.e., $\max(|x_1 - x_2|, |y_1 - y_2|, |z_1 - z_2|) \leq 1$. In our work, the voxels/3-cells constituting the object A are connected in 26N [14].

To derive the orthogonal cover of A , we define $\mathbb{G} := (\mathbb{G}_{yz}, \mathbb{G}_{zx}, \mathbb{G}_{xy})$ as the underlying *grid*. It consists of three orthogonal sets of equi-spaced *grid lines*, namely \mathbb{G}_{yz} , \mathbb{G}_{zx} , and \mathbb{G}_{xy} , their respective grid lines being perpendicular to yz -, zx -, and xy -planes (Fig. 1). The distance between the two consecutive grid lines of \mathbb{G}_{yz} (\mathbb{G}_{zx} or \mathbb{G}_{xy}) is defined as the *grid size*, g , which is a positive integer. The point of intersection of three orthogonal grid lines is termed as the *grid point*. Observe that for $g = 1$, the grid \mathbb{G} essentially corresponds to \mathbb{Z}^3 , as shown in Fig. 1. Further, as each grid point p is equivalent to a 3-cell c_p centered at p for $g = 1$, each face of c_p is a *grid face* lying on a *grid plane*, which is parallel to one of the three coordinates planes. In particular, all the six faces of any 3-cell lie in the grid planes which are parallel to coordinate planes and are unit distance apart. For $g > 1$, we have cubes of length g each. Each such cube is called a *unit grid cube* (UGC) whose vertices are *grid vertices*, edges constituted by *grid edges*, and faces constituted by grid faces. Each face of a UGC lies on a *grid plane*, which is parallel to one of three coordinates planes. Clearly, the distance of each grid plane from its parallel coordinate plane is an integer multiple of g . A smaller (larger) value of g implies a denser (sparser) grid. For notational simplicity, henceforth we use the notation \mathbb{G} to represent both the grid-model and the cell-model, the meaning being evident from the context.

The *orthogonal cover* of the object A is defined as an orthogonal polytope $P(A, \mathbb{G})$ that tightly circumscribes A . An *orthogonal polytope* is a simple polytope (i.e., each vertex is incident on exactly three edges) with all its vertices as grid vertices, all its edges made of grid edges, and all its faces lying on grid

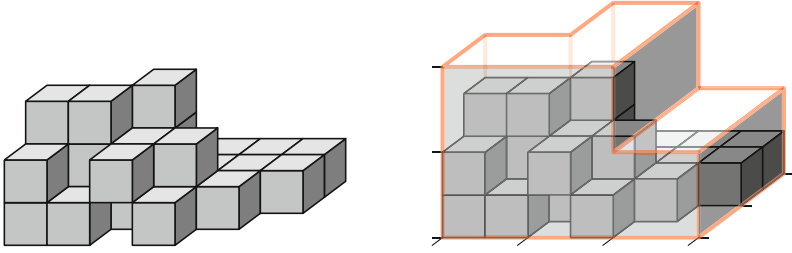


Fig. 2. An object A in \mathbb{Z}^3 and its corresponding orthogonal cover for $g = 2$

planes. Each face of an orthogonal polytope is an isothetic polygon whose alternate edges are orthogonal and constituted by grid edges of \mathbb{G} .

Problem definition. Given the 3D object A imposed on the grid \mathbb{G} , the problem is to construct its orthogonal polytope $P(A, \mathbb{G})$, such that the following conditions are satisfied:

- each point $p \in A$ lies inside $P(A, \mathbb{G})$;
- each vertex of $P(A, \mathbb{G})$ is a grid vertex;
- each edge of $P(A, \mathbb{G})$ is parallel to one of the coordinate axes;
- each face of $P(A, \mathbb{G})$ lies on some grid plane;
- volume of $P(A, \mathbb{G})$ is minimized.

Data structure. A *doubly connected edge list* (DCEL) is a data structure that stores topological information about a 2D subdivision (possibly embedded in 3D space) as a collection of the following records: a) vertex list, b) edge list, and c) face list. The vertex list contains all the vertices of the polytope excluding duplicates. The edge list has all the edges of the polytope in sets of four edges per face of UGC. Each edge is stored as a half-edge (mentioned below) represented by its source vertex and destination vertex. Four consecutive half-edges are assigned the face number representing the face to which the edges belong. For each half-edge $e_{ij} \in f_i$ in the edge list, the plane (yz , zx , or xy) to which the corresponding face f_i is parallel, is also recorded. The edge list also records the pairing of all half-edges (Sec. 3.1). The face list stores the id of a half-edge for each face of the polytope. This half-edge is considered as the first half-edge from which the face can be traversed by referring to the previous and next pointers in the edge list [11, 18].

3 Proposed Algorithm

The algorithm `ORTHOCOVER3D` first computes the start vertex v_s of $P(A, \mathbb{G})$ from the top-left-front point $p_0(i_0, j_0, k_0)$ of A . The object A is stored in a 3D array in which ‘1’s and ‘0’s represent object points and background points, respectively. The grid \mathbb{G} is also represented as a 3D array of structures; each

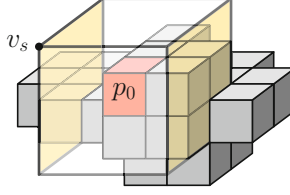


Fig. 3. Defining the start vertex, v_s , from the top-left-front point p_0 (red) of A

structure contains complete information about a UGC—complete and ordered in all respect regarding its eight vertices, twelve edges, and six faces. Observe that p_0 is the top-left-front point of A if and only if for each other digital point $p(i, j, k) \in A$, we have $(k < k_0) \vee ((k = k_0) \wedge (i > i_0)) \vee ((k = 0) \wedge (i = 0) \wedge (j < j_0))$. The point p_0 always lies strictly inside a UGC, whether lying on a grid line, or coinciding with a grid point, or be an ordinary (i.e., non-grid) digital point. From p_0 , the coordinates of v_s are obtained as

$$i_s = \lfloor i_0/g \rfloor \times g, \quad j_s = \lfloor j_0/g \rfloor \times g, \quad k_s = \lfloor k_0/g \rfloor \times g. \quad (1)$$

After obtaining v_s , one of the eligible UGC-faces (front face) having v_s as a vertex is enqueued in a queue, Q . Iteratively, each UGC-face f_i is dequeued from Q , and the faces incident on all edges of f_i are checked one by one for their eligibility of belonging to the orthogonal cover, using the `ELIGIBLEFACE` procedure. If such a UGC-face is eligible as a part of a polygonal face of the orthogonal cover, then it is enqueued only once in Q . After all the UGC-faces lying close to the object surface are considered, the faces are merged by `MERGEFACE` to derive the actual faces of the cover as isothetic polygons parallel to yz -, zx -, and xy -planes. The steps of the algorithm and its related procedures are given below.

Notations: $n_f = \# \text{faces}$, $n_e = \# \text{edges}$, $n_v = \# \text{vertices}$; $fid[\cdot]$, $eid[\cdot]$, and $vid[\cdot]$ denote face id, edge id, and vertex id; $start[e]$ denotes the start vertex of an edge e ; e_{ij} denotes the j th ($j = 1, 2, 3, 4$) edge of face f_i , and its paired half-edge in f_k (if coplanar with f_i) is denoted by \bar{e}_{ij} .

Algorithm. `ORTHOCOVER3D(A, G)`

01. $v_s \leftarrow (i_s = \lfloor i_0/g \rfloor \times g, j_s = \lfloor j_0/g \rfloor \times g, k_s = \lfloor k_0/g \rfloor \times g)$
02. pick the front face f_s of the UGC having v_s as a vertex
03. $n_f \leftarrow n_e \leftarrow n_v \leftarrow 0$
04. $fid[f_s] \leftarrow n_f \leftarrow n_f + 1$
05. $color[f_s] \leftarrow \text{GRAY}$
06. `ENQUEUE(Q, f_s)`
07. **while** Q is not empty
08. $f_i \leftarrow \text{DEQUEUE}(Q)$
09. **for** each edge $e_{ij} \in f_i$
10. $eid[e_{ij}] \leftarrow n_e \leftarrow n_e + 1$


```

11.   face[eij] ← fid[fi]
12.   if start[eij] ∉ V
13.     vid[start[eij]] ← nv ← nv + 1
14.     V ← V ∪ {start[eij]}
15.   assign start[eij], next[eij], prev[eij] ▷ from array G
16.   pair[eij] ← eid[eij]
17.   E ← E ∪ {eij}
18.   for each face fk incident on (start[eij], start[next[eij]])
19.     if ELIGIBLEFACE(fk) = TRUE
20.       if color[fk] = WHITE
21.         fid[fk] ← nf ← nf + 1
22.         color[fk] ← GRAY
23.         EnQueue(Q, fk)
24.       else if color[fk] = BLACK and fi and fk are coplanar
25.         pair[eij] ← eid[eij] ▷ eij ∈ fk
26.   edge[fi] ← eid[ei1] ▷ ei1 is the 1st edge of fi
27.   color[fi] ← BLACK
28.   F ← F ∪ {fi}
29. MERGEFACE(F, E)

```

Procedure. ELIGIBLEFACE(*f_k*)

```

01. var ← var2 ← FALSE
02. if fk intersects A
03.   return FALSE
04. else if there exists a point p of A in UGC1
05.   var1 ← TRUE ▷ UGC1 is the left UGC of fk
06. else if there exists a point p of A in UGC2
07.   var2 ← TRUE ▷ UGC2 is the right UGC of fk
08. if (var1 = TRUE and var2 = FALSE) or (var1 = FALSE and var2 = TRUE)
09.   return TRUE

```

Procedure. MERGEFACE(*F*, *E*)

```

01. for each face fi ∈ F
02.   eij ← edge[fi] ▷ 1st edge of fi ∈ F
03.   count ← 4 ▷ number of edges to be visited for fi
04.   do
05.     if eid[eij] ≠ pair[eij] ▷ pair[eij] exists
06.       eij ← pair[eij]
07.       fk ← face[eij]
08.       next[prev[eij]] ← next[eij]
09.       next[prev[eij]] ← next[eij]
10.       et ← next[prev[eij]]
11.       face[eij] ← fid[fi]
12.       F ← F ∖ {fk}

```

```

13.       $E \leftarrow E \setminus \{e_{ij}, \bar{e}_{ij}\}$ 
14.       $e_{ij} \leftarrow e_t$ 
15.       $edge[f_i] \leftarrow eid[e_{ij}]$   $\triangleright$  1st edge of the modified face  $f_i$ 
16.       $count \leftarrow count + 2$   $\triangleright$  4 edges added to & 2 edges removed from  $f_i$ 
17.      else  $\triangleright$   $pair[e_{ij}]$  does not exist
18.       $e_{ij} \leftarrow next[e_{ij}]$ 
19.       $count \leftarrow count - 1$ 
20.      while  $count \neq 0$ 
21. return  $(F, E)$ 

```

3.1 DCEL Construction

The process of constructing the DCEL starts with initialization of all the UGCs of \mathbb{G} (e.g., setting the color of each grid face of \mathbb{G} as WHITE, marking each grid edge as not in E , and marking each grid vertex as not in V). A UGC face is enqueued in Q only once after setting its color to GRAY and assigning it a unique id (Steps 4-6 and Steps 20-23 of ORTHOCOVER3D). Once a face f_i is dequeued from Q after all the faces incident at the edges of f_i have been considered for eligibility, the face f_i is marked as BLACK. The UGC-faces are thus traversed by BFS, as shown in Steps 7-28. For each edge $e_{ij} \in f_i$, each face f_k , incident on e_{ij} , is checked for its eligibility (Step 19). An eligible face is enqueued only when its color is WHITE. Otherwise, if f_k is BLACK and coplanar with f_i , then f_k can be merged with f_i by deleting their common edge, e_{ij} ; hence, $pair[e_{ij}]$ is set to the id of the edge $\bar{e}_{ij} \in f_k$ whose start and end vertices are the respective end and start vertices of e_{ij} (Steps 24-25).

For each face in F , we maintain its fid , the id of its incident edge, and the corresponding coordinate plane to which it is parallel. All these attributes are obtained and updated as shown in different steps of the algorithm ORTHOCOVER3D (Steps 4, 21, and 26). The edge information includes the edge id, the face on which it is incident, its start vertex, ids of its next and previous edges, and its paired half-edge, which are updated in Steps 10, 11, 15, 16, and 25. Vertex information consists of the vertex id, and its coordinates obtained from \mathbb{G} , as shown in Steps 12-14. A vertex is included in the vertex set exactly once (Step 12), which is ensured by setting a flag against each vertex recorded in the concerned structure of its UGC maintained in the 3D array corresponding to \mathbb{G} , as explained earlier.

The procedure ELIGIBLEFACE checks whether a grid face f_k is eligible for being a face of the 3D orthogonal cover. First, Step 1 initializes two boolean variables, namely $var1$ and $var2$, to FALSE. Step 2 checks whether the grid face f_k contains an object voxel (3-cell). If so, then f_k is not eligible (Step 3; see also Fig. 4(a)). In Step 4, it is checked whether any point $p \in UGC_1$ is an object voxel. If true, then the flag $var1$ is set to TRUE. Note that UGC_1 denotes the unit grid cube lying in between f_k and f_{k-g} , where f_k and f_{k-g} are two parallel consecutive grid faces separated by a distance of g . Similarly, UGC_2 is the unit grid cube lying in between f_k and f_{k+g} . In Step 6, similar checking is done for the set of points inside UGC_2 to find whether any of them belongs to the object.

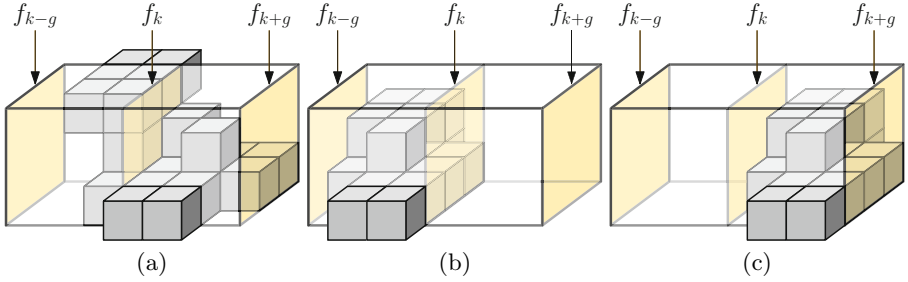


Fig. 4. Testing the eligibility of a face ($g = 3$): (a) f_k intersects the object A ; (b) The UGC between f_{k-g} and f_k has object occupancy but no object point is on f_k ; (c) The UGC between f_k and f_{k+g} has object occupancy but no object point is on f_k

If no point p in this set belongs to the object then *var2* remains FALSE. Finally, in Step 8 we ensure that if exactly one adjacent UGC contains object point(s), then only f_k is an eligible face of the orthogonal cover.

3.2 Face Merging

Once the BFS is over, the face list F contains all the grid faces of those UGCs which contain the points on the surface of the object. The MERGEFACE procedure considers each face f_i listed in F (Step 1) and merges all the adjacent faces that are coplanar. We start from the edge e_{ij} listed in F corresponding to f_i , and iteratively follow its next pointer in E to merge the faces. The **do while** loop iterates until the traversal reaches the starting edge corresponding to f_i (Steps 4-21). The variable *count* is used to keep track of the number of edges that are yet to be traversed—as the face merging progresses—to construct a single face as an isothetic polygon of maximal size out of all contiguous UGC-faces that are coplanar with f_i . When *count* = 0 (Step 20), no other face can be merged with the current face f_i , and the next face is considered. If there is a pair of e_{ij} , namely \bar{e}_{ij} , such that $pair[e_{ij}] = eid[\bar{e}_{ij}] \neq eid[e_{ij}]$, then it indicates that f_k , the face corresponding to \bar{e}_{ij} , is coplanar with f_i (Step 5); hence the face f_k is merged with f_i by deleting e_{ij} , its pair \bar{e}_{ij} , and the face f_k (Steps 12-13), and by readjusting the pointers of the related edges (Steps 8-9). Finally, MERGEFACE returns the face list F and the edge list E , where some of the faces have been merged and the first edges of the merged faces have been modified accordingly.

Figure 5 shows a simple example how the grid faces are merged. Let f_1 be selected first from F and e_1 is its start edge, with initialized *count* = 4. As e_1 does not have a paired half-edge whose face is coplanar with f_1 , it remains an edge of the face polygon, and the next edge e_2 is considered (*count* = 3). The edge e_2 also does not have a pair (*count* = 2), but its next edge e_3 has a pair, e_5 . Hence, e_3 and e_5 are deleted, and the faces f_1 and f_5 are merged, i.e., f_5 is deleted from F , $next[e_2]$ is set to e_6 , $next[e_8]$ set to e_4 , the face number of all the remaining edges of f_5 are set to $fid[f_1]$, and *count* increases to $2 + 2 = 4$ (Fig. 5(a,b)). In a similar way, as e_6 has e_{12} as its pair, the newly merged face is again merged

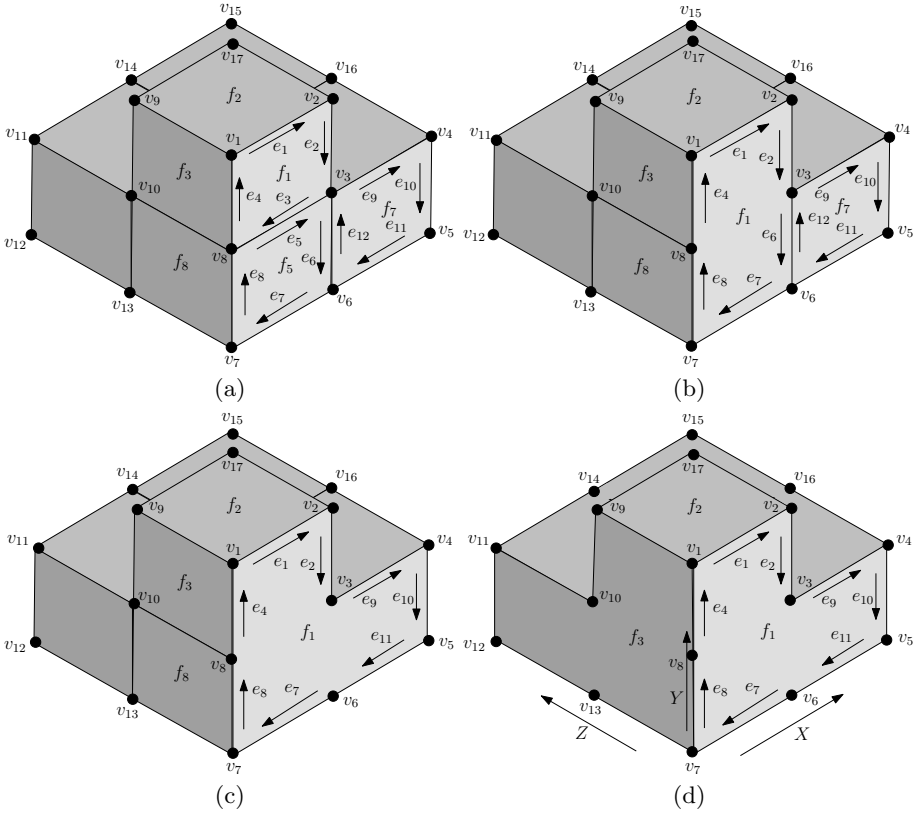


Fig. 5. Demonstration of face merging

with f_7 , the face corresponding to the half-edge e_{12} . The edges e_6 and e_{12} are deleted from E , f_7 deleted from F , $next[e_2]$ is set to e_9 , $next[e_{11}]$ set to e_7 , and $count$ increases further to $4 + 2 = 6$ (Fig. 5(b,c)). As the traversal continues, each of e_9 , e_{10} , e_{11} , e_7 , e_8 , and e_4 having no pairing half-edge, are included as edges of the orthogonal cover; $count$ falls to $6 - 6 = 0$, and the traversal stops, as e_1 is reached. Thus an orthogonal polygon, $v_1v_2v_3v_4v_5v_6v_7v_8v_1$, is obtained parallel to the xy -plane (Fig. 5(c,d)). Figure 5(d) shows all the merged faces in different planes for the simple orthogonal polytope.

3.3 Time Complexity

We disregard the time required for grid initialization. The **while** loop of the algorithm ORTHOCOVER3D (Steps 7-28) is executed once for each face on the cover. A face f_i is added to Q only if it is an eligible face and not yet been enqueued (Steps 9-11). If enqueued, the face f_i is marked as visited (i.e., colored GRAY, Steps 19-23). So, the while loop runs for the number of times equal to the total number of grid faces constituting the orthogonal cover. Let n be

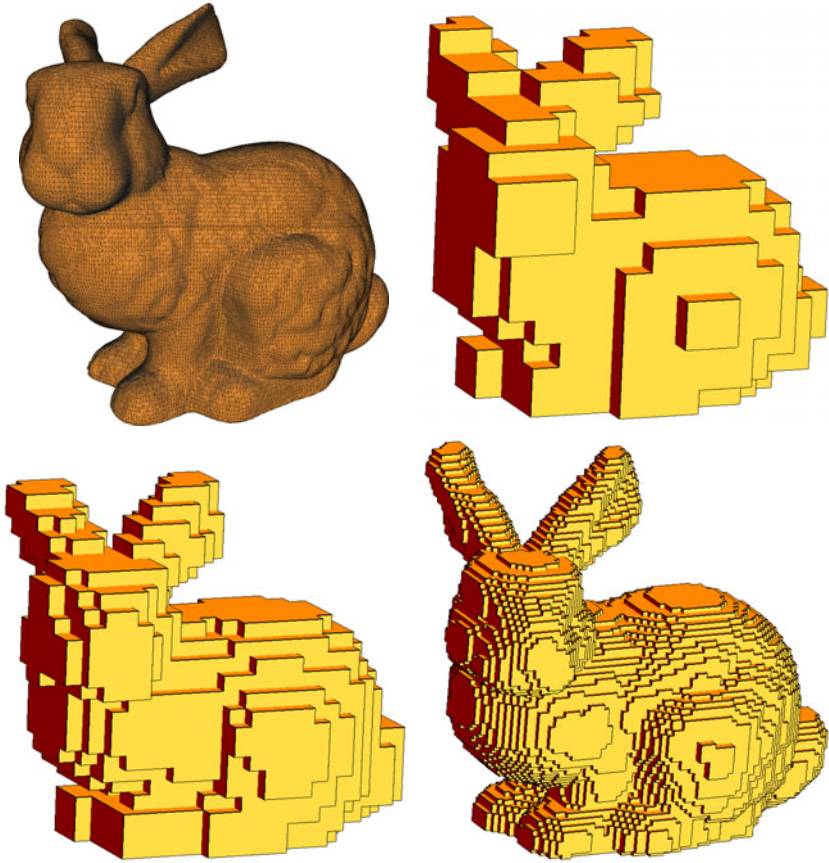


Fig. 6. “Stanford Bunny” and its covers for different grid sizes. Top-left: original object, $n = 58605$. Top-right: $g = 12$, $n_v = 769$, $n_e = 1492$, $n_f = 367$. Bottom-left: $g = 7$, $n_v = 2040$, $n_e = 3864$, $n_f = 903$. Bottom-right: $g = 2$, $n_v = 22305$, $n_e = 42896$, $n_f = 10377$.

the number of voxels constituting the object surface connected in $26N$. Then the runtime complexities for the best and the worst cases can be analyzed as follows.

Best Case: Minimum number of UGCs, each UGC being a cube of length g and containing points of the object surface, is $O(n/g^3)$. As each UGC has six faces—a maximum of five of which can be a part of the cover, the number of grid faces on the surface of the object is again $O(n/g^3)$. In each iteration, the eligibility of a grid face f_k for being a face on the orthogonal cover is decided by checking the voxels inside the two UGCs on both sides of (i.e., adjacent to) f_k , which requires $O(1)$ comparisons in the best case (Step 19). So, the **while** loop (Steps 7-28) over all the grid faces comprising the orthogonal cover requires at most $O(n/g^3) \times O(1) = O(n/g^3)$ computations. To check the color of a face, or to check whether a vertex is already included in V , or to check an edge is

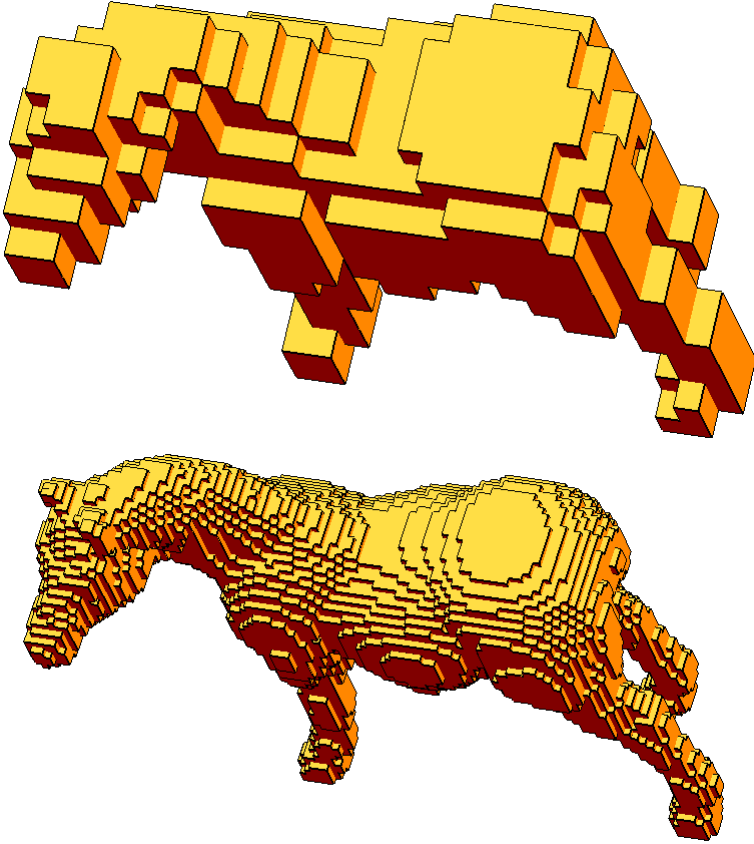


Fig. 7. Results on “Horse”. Top: $g = 8, n_v = 1205, n_e = 2328, n_f = 523$. Bottom: $g = 2, n_v = 25512, n_e = 53188, n_f = 13238$.

already in E , we use the grid information as explained earlier. Hence, each of these checks (Steps 12, 20, 24) needs constant time.

Before applying the MERGEFACE procedure, the total number of edges (half-edges) in E is four times the total number of eligible faces, i.e., $4 \times O(n/g^3) = O(n/g^3)$; the total number of vertices is also $O(n/g^3)$, as each vertex is incident on at most six edges. During merging, the **for** loop (Steps 1-20) considers each face $f_i \in F$ one by one (**do-while** loop: Steps 4-20), and checks the pair of e_{ij} . If $pair[e_{ij}] \neq eid[e_{ij}]$, then it indicates that the face f_k on which $pair[e_{ij}]$ is incident, is coplanar with f_i . So the face f_k is merged with f_i and necessary updates/modifications of the relevant edges and faces are done in constant time. Since the total complexity (number of faces, edges, and vertices) of the DCEL is bounded by $O(n/g^3)$, the time complexity of MERGEFACE becomes $O(n/g^3)$. So, the best-case time complexity of the algorithm ORTHOCOVER3D is given by $O(n/g^3)$.

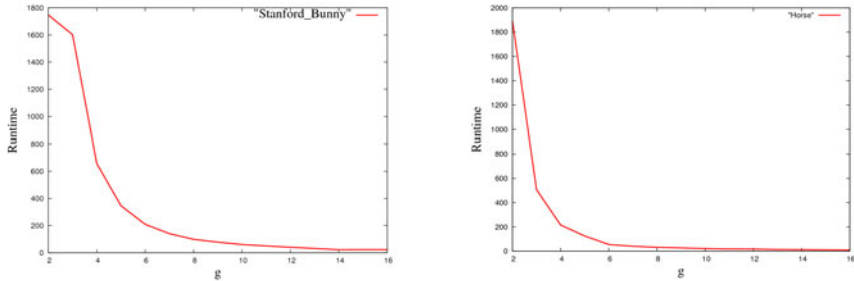


Fig. 8. Runtime (in secs.) versus grid size g . Left: Stanford Bunny. Right: Horse

Worst Case: Each UGC may contain as low as $O(g)$ surface voxels, and so each grid face may require $O(g^3) - O(g) = O(g^3)$ time for checking its eligibility. Maximum number of UGCs can be $O(n/g)$, wherefore we have $O(n/g) \times O(g^3) = O(n g^2)$ time complexity in the worst case for finding the faces of the orthogonal cover. Maximum number of faces would be $O(n/g)$, since each UGC (containing surface voxels) will contribute at least one face and at most five faces to the orthogonal cover. Hence, with a justification similar to the best-case analysis, the time complexity of MERGEFACE is $O(n/g)$. The overall worst-case time complexity is, therefore, given by $O(n g^2) + O(n/g) = O(n g^2)$.

In practice, however, we find that the runtime decreases rapidly with increasing grid size. This is revealed by our exhaustive experimentation, some of which are presented in Sec. 4. Thus, actual runtime for real-world digital objects in \mathbb{Z}^3 tends towards the best case, which is a characteristic of our algorithm.

4 Results and Conclusion

We have implemented the algorithm in C in Linux Fedora Release 7, Kernel version 2.6.21.1.3194.fc7, Dual Intel Xeon Processor 2.8 GHz, 800 MHz FSB. OpenGL running with MinGW on Windows XP Professional is used for the purpose of 3D rendering [11][12]. The algorithm has been tested on several 3D objects for different grid sizes, a few of them being presented in Figs. 6 and 7.

Notice that as g is decreased in Fig. 6, a tighter description of the bunny is obtained, as the numbers of vertices, edges, and faces increase rapidly. Figure 8 shows the runtimes of the algorithm on different objects (“Stanford Bunny” and “Horse”) for different grid sizes. For $g = 2, 8,$ and 16 , the respective CPU times required to construct the orthogonal covers for “Stanford Bunny” are 1747, 98, and 22 seconds, whereas those for “Horse” are 1893, 32, and 10 seconds. Clearly, the runtime falls significantly with the increase in grid size. The description of the object in the form of the orthogonal cover at a lower grid size contains more information (than the one at a higher grid size) about the nature of the object at the cost of more CPU time and storage space.

References

1. Berg, M.D., Cheong, O., Kreveld, M.V., Overmars, M.: *Computational Geometry—Algorithms and Applications*. Springer, Heidelberg (1997)
2. Bhaniramka, P., Wenger, R., Crawfis, R.: Isosurface construction in any dimension using convex hulls. *IEEE Trans. on Visualization and Computer Graphics* 10, 130–141 (2004)
3. Bhaniramka, P., Wenger, R., Crawfis, R.: Isosurfacing in higher dimensions. In: *Proceedings of Visualization*, Salt Lake City, pp. 267–273 (2000)
4. Biswas, A., Bhowmick, P., Bhattacharya, B.B.: Construction of Isothetic Covers of a Digital Object: A Combinatorial Approach. *Journal of Visual Communication and Image Representation* 21, 295–310 (2010)
5. Biswas, A., Bhowmick, P., Bhattacharya, B.B.: TIPS: On Finding a Tight Isothetic Polygonal Shape Covering a 2D Object. In: Kalviainen, H., Parkkinen, J., Kaarna, A. (eds.) *SCIA 2005*. LNCS, vol. 3540, pp. 930–939. Springer, Heidelberg (2005)
6. Brimkov, V.: Discrete volume polyhedrization: Complexity and bounds on performance. In: *Proc. of the International Symposium on Computational Methodology of Objects Represented in Images: Fundamentals, Methods and Applications*, CompIMAGE 2006, pp. 117–122. Taylor and Francis, Coimbra (2006)
7. Coeurjolly, D., Sivignon, I.: Reversible discrete volume polyhedrization using Marching Cubes simplification. In: *SPIE Vision Geometry XII*, vol. 5300, pp. 1–11 (2004)
8. Cohen-Or, D., Shamir, A., Shapira, L.: Consistent Mesh Partitioning and Skeletonization using the Shape Diameter Function. *The Visual Computer* 24, 249–259 (2008)
9. Giles, M., Haines, R.: Advanced interactive visualization for CFD. *Computing Systems in Engineering* 1, 51–62 (1990)
10. Golovinskiy, A., Funkhouser, T.: Randomized cuts for 3D mesh analysis. *ACM Transactions on Graphics (Proc. SIGGRAPH ASIA)* 27, Article 145 (2008)
11. Hearn, D., Baker, M.P.: *Computer Graphics with OpenGL*. Pearson Education Inc., London (2004)
12. Hill, F.S., Kelley, S.M.: *Computer Graphics Using OpenGL*. Pearson Education Inc., London (2007)
13. Katz, S., Leifman, G., Tal, A.: Mesh segmentation using feature point and core extraction. *The Visual Computer* 21, 649–658 (2005)
14. Klette, R., Rosenfeld, A.: *Digital Geometry: Geometric Methods for Digital Picture Analysis*. Morgan Kaufmann, San Francisco (2004)
15. Livnat, Y., Shen, H.-W., Johnson, C.: A near optimal isosurface extraction algorithm using span space. *IEEE Trans. Visualization and Computer Graphics* 2, 73–84 (1996)
16. Lorensen, W.E., Cline, H.E.: Marching Cubes: A high resolution 3D surface construction algorithm. *Computer Graphics* 21, 163–169 (1987)
17. Newman, T.S., Yi, H.: A Survey of the Marching Cubes Algorithm. *Computers & Graphics* 30, 854–879 (2006)
18. Preparata, F.P., Shamos, M.I.: *Computational Geometry—An Introduction*. Springer, New York (1985)
19. Shapira, L., Shalom, S., Shamir, A., Cohen-Or, D., Zhang, H.: Contextual Part Analogies in 3D Objects. *International Journal of Computer Vision* 89, 309–326 (2010)

20. Shlafman, S., Tal, A., Katz, S.: Metamorphosis of Polyhedral Surfaces using Decomposition. In: Eurographics 2002, pp. 219–228 (2002)
21. Turk, G., Levoy, M.: Zippered polygon meshes from range images. In: SIGGRAPH 1994, pp. 311–318 (1994)
22. Weber, G., Kreylos, O., Ligocki, T., Shalf, J., Hagen, H., Hamann, B.: Extraction of crack-free isosurfaces from adaptive mesh refinement data. In: Proceedings of VisSym 2001, Ascona, Switzerland, pp. 25–34 (2001)
23. Wilhelms, J., van Gelder, A.: Topological considerations in isosurface generation extended abstract. *Computers Graphics* 24, 79–86 (1990)

Skeleton Path Based Approach for Nonrigid 3D Shape Analysis and Retrieval

Chunyuan Li and A. Ben Hamza

Concordia Institute for Information Systems Engineering
Concordia University, Montréal, QC, Canada

Abstract. In this paper, we propose a skeleton path based approach to analyze and retrieve nonrigid 3D shapes. The main idea is to match skeleton graphs by comparing the geodesic paths between skeleton endpoints. Our approach is motivated by the fact that the path feature is stable in the presence of articulation of components. The experimental results demonstrate the performance of our proposed method in terms of robustness to symmetry, discrimination against different graph structures, and high efficiency in nonrigid shape retrieval.

Keywords: Skeleton Path, Nonrigid 3D Shapes, Retrieval.

1 Introduction

With the increase in the number of scanned 3D objects, 3D shape analysis and retrieval is becoming popular in the fields of computer vision, computer graphics, and computer aided design. Previous efforts have been, however, mainly devoted to rigid 3D models, and thus how to efficiently and effectively analyze and compare nonrigid shapes is still a challenging problem.

The curve skeleton, which integrates geometrical and topological features of the object, is an important shape descriptor. Shape similarity based on skeleton matching usually performs better than mesh surface or other shape descriptors in the presence of articulation of components, especially for non-rigid shape. As pointed out in [7], the curve-skeleton provides characteristics like part/component matching, registration and visualization, intuitiveness, and articulated transformation invariance.

Nonrigid shape matching is one of the most challenging problems in content-based 3D object retrieval. The aim of the SHREC 2010 –Shape Retrieval Contest of Non-rigid 3D Models– is to evaluate and compare the effectiveness of different 3D retrieval methods [10]. For a 3D retrieval algorithm, the shape descriptor and the similarity discrimination are two key components. The global and local isometry-invariant descriptor proposed recently by Wu *et al.* [18] captured well the global and local information. Also, Agathos *et al.* [1] proposed a retrieval methodology based on a graph-based object representation. This method makes use of a meaningful new mesh segmentation and the Earth mover’s distance (EMD) similarity measure.

In recent years, several skeleton based shape analysis and retrieval methods have been proposed. Sundar *et al.* [16] encoded the geometric and topological information in the form of a skeleton graph and used graph matching techniques for skeleton matching and comparison. Cornea *et al.* [7] enhanced the framework by using a new skeletonization algorithm and an extension of the many-to-many matching algorithm. Au *et al.* [2] presented a fast and fully automatic correspondence algorithm that allows matching of a wide variety of shapes with semantically similar structures but with different geometric details. Specifically, they attempted to find a one-to-one semantic correspondence between two sets of feature nodes of curve skeletons. However, determining the similarity between two given shapes does not necessarily require finding an exact correspondence between their shape components. Our work is partly a 3D extension of the 2D path similarity skeleton graph matching approach proposed by Bai. *et al* [4]. Unlike [4], a major goal of our approach is to discover the symmetry instead of finding correspondences. Siddiqi *et al.* [15] introduced a medial surface based method and obtained state-of-the-art performance on McGill Articulated Shape Benchmark [15].

Partly motivated by Leonardo’s Vitruvian Man, which describes the perfect human form in geometrical terms, we propose a skeleton path feature to represent each component of a non-rigid 3D shape, assuming that this feature descriptor is isometry-invariant, i.e. invariant to the object’s variational representation, rotation, translation, scaling, and nonrigid bending. A skeleton path refers to the geodesic paths between two endpoints in the curve-skeleton, as shown in Fig. 1(a), where these shortest paths are represented as sequences of radii of the maximal balls at the corresponding skeleton points. We also benefit from the fact that the proportions of the curve skeleton length for different components are different and are almost constant. Although we do not explicitly consider the topological structure of the skeleton graphs, we do not, however, completely ignore this structure. It is worth pointing out that this topological structure is implicitly represented by the fact that overlapping parts of the geodesic paths are similar. Therefore, our approach is flexible enough to perform extremely well on nonrigid 3D shapes.

The rest of the paper is organized as follows. Section 2 describes the proposed approach. The experimental results using the proposed algorithm are provided in Section 3. Finally, we conclude in Section 4.

2 Proposed Approach

Our proposed method may be described as a two-phase approach: 1) Skeleton path acquisition, which includes curve-skeleton extraction, endpoint detection, and path construction. 2) Endpoints matching, which consists of finding an ordered sequence of end nodes and post-processing of this new sequence.

For convenience and efficiency, we adopt the curve-skeleton extraction algorithm developed by Cornea *et al.* [8]. Nevertheless, our algorithm can be generally applied on curve-skeletons with satisfactory homotopic and centered properties [9].

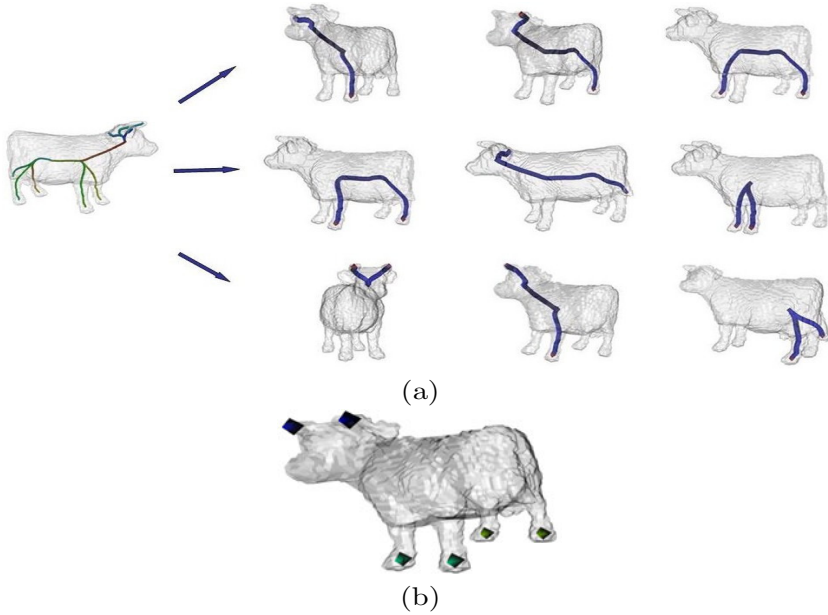


Fig. 1. (a) Our skeleton path based algorithm describes a 3D cow model as many paths between end nodes. (b) Symmetric components that are discovered by our algorithm are represented by end nodes rendered with the same colors. Note that only part of the results is shown.

The proposed curve-skeleton extraction algorithm works on a volumetric representation of a 3D object. It uses a generalized potential field generated by charges placed on the surface of the object. Given a 3D vector field, we use concepts from vector field visualization to identify two types of seed points that we will use to construct a curve-skeleton: critical points and high divergence points. Skeleton segments are discovered using a force-following algorithm on the underlying vector field, starting at each of the identified seed points. The force-following process evaluates the vector (force) value at the current point and moves in the direction of the vector with a small pre-defined step. At critical points, where the force vanishes, the initial directions are determined by evaluating the eigenvalues and eigenvectors of the Jacobian at the critical point. More details about computing the curve-skeleton can be found in [8].

2.1 Skeleton Path

We first describe the initial steps for building the skeleton graphs. The following definitions apply to continuous skeletons as well as to curve-skeletons of 3D models (composed of voxels).

Definition 1. A skeleton point having only one adjacent point is an endpoint (the skeleton endpoint); a skeleton point having three or more adjacent points is a junction point. If a skeleton point is not an endpoint or a junction point, it is called a connection point.

Definition 2. The endpoint in the skeleton graph is called an end node, and the junction point in the skeleton graph is called a junction node.

Definition 3. The shortest path between a pair of end nodes on a skeleton graph is called a skeleton path.

Based on the curve skeletons that are extracted using the method described above, we provide details on how we detect endpoints and construct the skeleton path.

A. Skeleton Endpoint Detection

Since the curve-skeleton consists of many segments that have two ends (i.e. segment endpoints), we detect the skeleton endpoints by considering the distance between these end nodes. We denote the set of all the N segment endpoints of an input skeleton by $P = \{p_1, p_2, \dots, p_N\}$. For simplicity, let $p_k \in P$ denote the testing endpoint. Given a threshold ε_{RD} , let $Q = \{q : \|p_k - q\| \leq \varepsilon_{RD}, q \in P, q \neq p_k\}$ be the nearest neighbor endpoint set. We consider p_k as a skeleton endpoint if the size of Q is 0 and as a junction point if the size is larger than 2. The rest of the segment endpoints and all the other skeleton points are connected points (see Fig. 2).

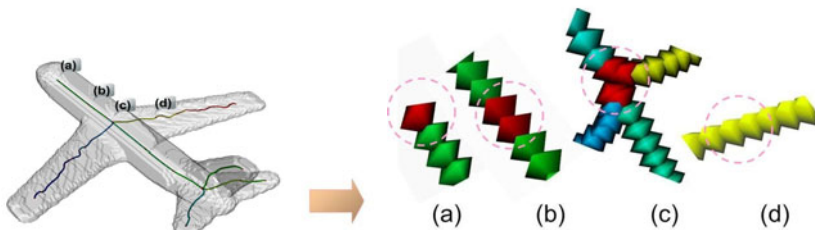


Fig. 2. Illustration of the critical skeleton points using our method on a 3D airplane (left). On the right, (a) endpoint, (b) connected points between two segments, (c) junction points, and (d) connected points in a segment. For better viewing, please see the original color pdf file.

B. Skeleton Path Construction

After endpoints are detected, we construct the skeleton path as the shortest path between two given nodes (see Fig. 1). Suppose there are N end nodes $\{v_i\}_{i=1, \dots, N}$ in the skeleton graph G to be matched.

Path Length Percentage. Let $\Gamma = (\gamma_{ij})$ be an $N \times N$ path length matrix, where γ_{ij} denotes the geodesic distance (shortest path length) from the i -th end

node \mathbf{v}_i to the j -th end node \mathbf{v}_j . To preserve scale-invariance, we normalize the matrix Γ by the overall curve skeleton length L_{skel} . In this way, we obtain an $N \times N$ length percentage matrix $L = (\ell_{ij})$, where $\ell_{ij} = \gamma_{ij}/L_{\text{skel}}$.

Path Radius Vector. Let $p(\mathbf{v}_m, \mathbf{v}_n)$ denote the skeleton path from \mathbf{v}_m to \mathbf{v}_n . We sample $p(\mathbf{v}_m, \mathbf{v}_n)$ with M equidistant points, which are all skeleton points. Let $R_{m,n}(t)$ be the radius of the maximal ball at the skeleton point with index t in $p(\mathbf{v}_m, \mathbf{v}_n)$. We define a vector of the radii of the maximal balls at the M sample points on $p(\mathbf{v}_m, \mathbf{v}_n)$ as follows:

$$R_{m,n} = (R_{m,n}(t))_{t=1,2,\dots,M} = (r_1, r_2, \dots, r_M). \quad (1)$$

The distance transform value for each point is equal to the radius of maximal inscribed ball. Suppose there are N_0 voxels in the original 3D model, then to make the proposed method invariant to scale, we normalize $R_{m,n}(t)$ in the following way:

$$R_{m,n} = \frac{DT(t)}{1/N_0 \sum_{i=1}^{N_0} DT(s_i)}, \quad (2)$$

where s_i varies over all N_0 voxels in the model [4].

Path Distance. The model dissimilarity between two skeleton paths is called a path distance. If $R = (r_i)_{i=1,\dots,M}$ and $R' = (r'_i)_{i=1,\dots,M}$ denote the vectors of radii of two model paths $p(\mathbf{u}, \mathbf{v})$ and $p(\mathbf{u}', \mathbf{v}')$ respectively, then the path distance is defined as

$$\varphi(p(\mathbf{u}, \mathbf{v}), p(\mathbf{u}', \mathbf{v}')) = \sum_{i=1}^M \frac{(r_i - r'_i)^2}{r_i + r'_i} + \alpha \frac{(\ell - \ell')^2}{\ell + \ell'}, \quad (3)$$

where ℓ and ℓ' are the length percentages of $p(\mathbf{u}, \mathbf{v})$ and $p(\mathbf{u}', \mathbf{v}')$ respectively. The parameter α is a weight factor [4].

In order to make our representation scale invariant, the path lengths are normalized. We include the path length percentages in Eq. (3), since the percentage is not reflected in the sequences of radii (all paths are sequences of M radii). Thus, our path representation and the path distance are scale invariant.

2.2 Endpoints Matching

Sorting endpoints by summing path length percentage. In a skeleton graph G with N end nodes $\{\mathbf{v}_i\}_{i=1,\dots,N}$, each end node has the skeleton paths to all other end nodes in the graph. Let $T(\mathbf{v}_i) = \sum_{j=1}^N \ell_{ij}$ be the total path length percentages of the end node \mathbf{v}_i . Thus, given an end node \mathbf{v}_k , there is a corresponding length percentage $T(\mathbf{v}_k)$. We order all the end nodes in G following these percentages, by ranking an endpoint with a higher percentage at the top of the list. Therefore, we obtain an ordered end node sequence $S = \{\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_N\}$.

Endpoints Distance. Let G and G' be two graphs to be matched, and denote by $S = \{\mathbf{u}_i\}_{i=1,\dots,K+1}$ and $S' = \{\mathbf{u}'_i\}_{i=1,\dots,N+1}$ their respective ordered end node sequences, with $K \leq N$. Similar to shape contexts [5], the matching cost $c(\mathbf{u}_i, \mathbf{u}'_j)$ between \mathbf{u}_i and \mathbf{u}'_j is based on the paths to all other end nodes in G and G' that emanate from \mathbf{u}_i and \mathbf{u}'_j , correspondingly. Then, we compute the path distances between the two sequences and obtain a matrix $\Phi = (\varphi(\mathbf{u}_i, \mathbf{u}'_j))$ of the path distances computed using Eq. (3). As suggested by Bai *et al.* [4], we use the optimal subsequence bijection (OSB) to compute the dissimilarity value:

$$c(\mathbf{u}_i, \mathbf{u}'_j) = OSB(\varphi(\mathbf{u}_i, \mathbf{u}'_j)) \quad (4)$$

Global Matching. Using Eq. (4), we compute the total dissimilarity matrix $\mathcal{C}(G, G') = (c(\mathbf{u}_i, \mathbf{u}'_j))_{\substack{1 \leq i \leq K+1 \\ 1 \leq j \leq N+1}}$ between G and G' using the Hungarian algorithm. For each end node \mathbf{v}_i in G , the Hungarian algorithm can find its corresponding end node \mathbf{v}'_i in G' . Since G and G' may have different numbers of end nodes, the total dissimilarity value should include a penalty for end nodes that did not find any partner. To achieve this, we simply add additional rows with a constant value κ so that $\mathcal{C}(G, G')$ becomes a square matrix. This constant value κ is the average of all the other values in $\mathcal{C}(G, G')$. The intuition behind using the Hungarian algorithm is that we want to have a globally consistent one-to-one assignment of all end nodes with possibly assigning some end nodes to κ , which represents a dummy node. This means that we seek a one-to-one correspondence of the end nodes in the skeleton graphs (with possibly skipping some nodes by assigning them to a dummy node). However, the Hungarian algorithm does not preserve the order of the matched sequences. This does not influence the final score, since we can change the order only for similar symmetric end nodes. This is also part of the reason why we can detect symmetric components.

3 Experimental Results

In this section, we evaluate the performance of the proposed method in three parts: Symmetric components discovery, matching between different graph structures, and illustration of the recognition performance of our method on McGill Articulated Shape Database.

3.1 Symmetric Components Discovery

Given an end node sequence $S = \{\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_N\}$ of a skeleton G , we obtain a new sequence \hat{S} by changing the order of certain end nodes in a set $C \subseteq G$ and compute the dissimilarity to the original. If the result $\mathcal{C}(S, \hat{S})$ is less than a given threshold ε_C , we consider the components containing these end nodes in C to be symmetric. Obviously, for a given nonrigid 3D shape, it is possible that there are more than one such set C , whose size may be larger than 2. In other words, there exist many symmetric component groups and that each group might have more

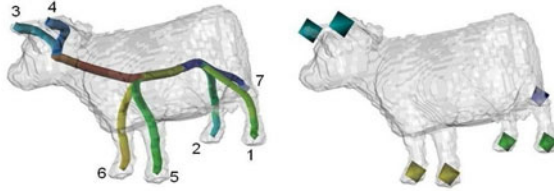


Fig. 3. The curve skeleton and discovered symmetric components indicated by end nodes with the same color

Table 1. The matrix of dissimilarity values between the skeleton graph with the corresponding end nodes exchanged and the original. The colored values are symmetric node pairs.

	1	2	3	4	5	6	7
1	0	0.0120	6.2616	2.7178	2.8752	2.0865	1.0400
2	0.0120	0	0.8897	3.3483	2.9786	1.7213	1.2096
3	6.2616	0.8897	0	0.0096	0.7720	2.0073	6.7315
4	2.7178	3.3483	0.0096	0	0.1581	5.0328	10.4491
5	2.8752	2.9786	0.7720	0.1581	0	0.0025	1.2911
6	2.0865	1.7213	2.0073	5.0328	0.0025	0	0.1677
7	1.0400	1.2096	6.7315	10.4491	1.2911	0.1677	0

than 2 components. We now give a simple example illustrating our symmetry components discovery approach. Fig. 3 shows the curve skeleton and the results on a 3D cow model. The end nodes, displayed with the same color, indicate the symmetric components. As we observe, the left front leg and the right front leg are symmetric components, they are shown in yellow color. The back legs in green and horns are displayed in blue. We indexed the end nodes so that symmetric nodes are clear to be presented. The matrix with elements indicating the dissimilarity if the corresponding nodes are exchanged is shown in Table 1. The dissimilarities between two most symmetric end nodes are marked with colored numbers. Here, we choose the parameters $\varepsilon_C = 0.1, M = 50, \alpha = 10$.

Besides the symmetry discovery of the 3D cow model in Fig. 3, we tested the process on several other examples. In Fig. 4(a), we first discover the symmetry in both the hands and the legs of a dancer. And the head is not symmetric to any part of the human body. Obviously, our method finds the correct symmetric components even in large variability due to articulation. Secondly, Fig. 4(b) shows the result of an eight-leg octopus, which demonstrates that the proposed

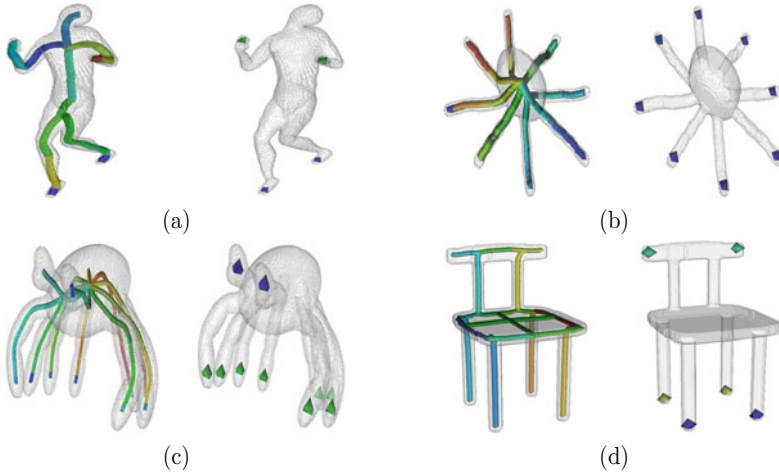


Fig. 4. (a) Symmetry discovery of a dancer; (b) Symmetry discovery of an octopus; (c) Symmetry discovery of a crab; (d) Symmetry discovery of a chair

method is able to discover symmetric groups with numerous members. A crab with eight legs and two eyes in Fig. 4(c) illustrates that our method works correctly in the situation when there are many symmetric groups with different quantities of member. Finally, we show the result of a four-leg chair in Fig. 4(d). It demonstrates that the proposed method also performs well in the presence of a rigid shape, even when its skeleton graph is not a tree.

3.2 Matching Skeletons with Different Graph Structures

For skeleton graphs with the same number of end nodes, they might have very different graph structure. Sometimes there are similar path radii vectors. But path length percentage will enhance the performance. For example, shapes like snake and spectacle have two endpoints in their curve-skeletons. Moreover, the skeleton graph of the snake is a tree, whereas the skeleton graph of the spectacle is not a tree. To evaluate the performance of our proposed algorithm on distinguishing the topological difference, we use a small database that contains four nonrigid 3D shapes: Two spectacles and two snakes as shown in Fig. 5. The parameter M for this database was set to $M = 50$. We also show the results with $\alpha = 0$ in Fig. 6(a) and $\alpha = 10$ in Fig. 6(b). By observing the rankings, it is evident that both of them could discriminate the skeleton graphs with different structures. Although the shortest paths between end nodes of the two classes are similar, the proposed method is, however, able to distinguish the structural difference between a closed loop and a line better by considering the length percentages. No matter how shapes, e.g. spectacles or snakes, deform due to articulation, the length percentages are always almost constant. Moreover, as



Fig. 5. Top: Two spectacles and their curve-skeletons; Bottom: Two snakes and their curve-skeletons

Query	1 st	2 nd	3 rd	Query	1 st	2 nd	3 rd
	 0.2527	 2.6444	 3.1303		 0.2527	 5.6741	 8.8446
	 0.2527	 2.3756	 2.6805		 0.2527	 5.0972	 7.5737
	 0.5088	 2.3756	 2.6444		 0.5523	 5.0972	 5.6741
	 0.5088	 2.6805	 3.1303		 0.5523	 7.5737	 8.8446

(a) (b)

Fig. 6. Comparison between $\alpha = 0$ and $\alpha = 10$ on a small database. The distance between query and the given shape is also displayed.

a matter of fact, shapes in different classes have different length percentages, which lead to more effective discrimination.

3.3 Retrieval on McGill 3D Articulated Shape Database

Based on the above experimental results, our algorithm is validated to be robust to symmetry and discriminative to different graph structures. We demonstrate it further on McGill Articulated Shape Database with 255 objects divided into ten categories, namely, ‘Ants’, ‘Crabs’, ‘Spectacles’, ‘Hands’, ‘Humans’, ‘Octopuses’, ‘Pliers’, ‘Snakes’, ‘Spiders’, and ‘Teddy Bears’. Sample models from this database are shown in Fig. 7

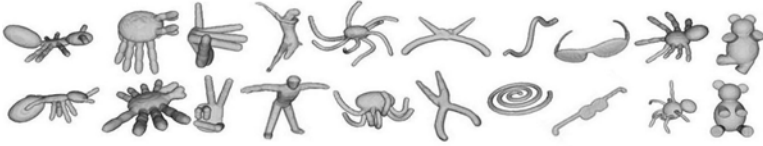


Fig. 7. Sample shapes from McGill Articulated Shape Database. Only two shapes for each of the 10 classes are shown.

Skeleton Path based Methods. Retrieving shapes that are similar to a given query shape from a database involves shape matching. However, determining the similarity between two given shapes does not necessarily require finding an exact correspondence between their shape components. In this section, we extend the shape similarity measure discussed in Section 3 to shape retrieval, and we also propose five methods based on the skeleton path. In the sequel, we will use the following abbreviations:

SH: We denote the method in section 3 as SH, since it uses the square matrix with penalty and Hungarian algorithm.

SDP: We denote the method that uses the square matrix with penalty and dynamic programming algorithm as SDP.

NSH: We denote the method that uses the matrix, which is not square and without penalty, and the Hungarian algorithm as NSH.

EMS: We define the dissimilarity from the query to a shape in the dataset as the sum of minimum endpoint distance of the query to all endpoints of the latter, and denote it as EMS.

PMS: We define the dissimilarity from the query to a shape in the dataset as the sum of minimum skeleton path distance of the query to all skeleton paths of the latter, and denote it as PMS.

Here we use the parameters $M = 50$ and $\alpha = 50$. In our comparative analysis, we have used the precision/recall curve to measure the retrieval performance. Ideally, this curve should be a horizontal line at unit precision. For each query shape, we use the first 77 returned shapes with descending similarity rankings (i.e., ascending Euclidean distance ranking), dividing them into 11 groups accordingly. The retrieval results of the 5 skeleton path based methods on the whole McGill Articulated Shape Database are shown in Fig. 8. Obviously, PMS provides a much better performance than the other methods because it fully exploits the original information that skeleton paths carry. By finding the minimum value of skeleton path distances, we might establish a potential corresponding relationship between paths. However, as a matter of fact, there is no veracious global path correspondence. As for EMS, the second best method, we assume that the endpoints with minimum distance are corresponding to each other, although it may fail to find the global endpoints correspondence. Furthermore, SH and SDP are almost neck and neck in terms of retrieval accuracy, and both are superior to NSH, which demonstrates that the penalty plays a key role in shape

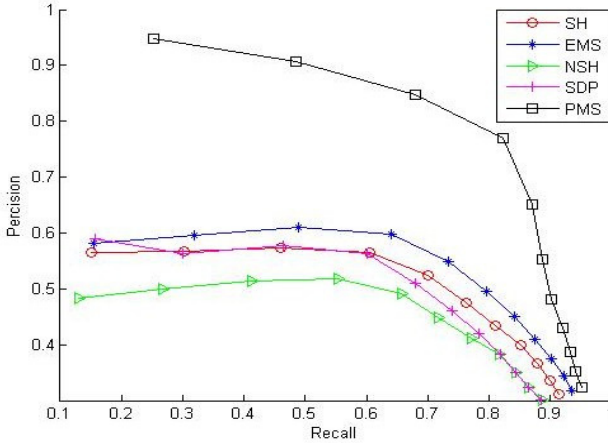


Fig. 8. Precision-recall plot of the proposed skeleton path based methods

discrimination. Our implementation was done in MATALB on a Intel Core 2 Duo with 2.0 GHz. To give an idea about the timing: constructing the skeleton path takes on average slightly less than 1 minute for a 3D model; most of this time is actually consumed by Dijkstra’s algorithm for finding the shortest path between two end nodes.

4 Conclusions

We proposed a skeleton path based technique that is able to detect symmetric components, discriminate different graph structure and retrieve nonrigid 3D shapes. We represented a nonrigid shape by a set of geodesic paths between skeleton endpoints. These paths were compared using sequence matching. By detecting symmetric components, our framework is shown to be consistent with human semanteme based on curve-skeleton. Also, we found that it is possible to discover multiple components in a symmetric group. In addition, the proposed approach could enhance the performance of distinguishing the topological difference. Finally, our skeleton path based approach is shown to be effective, efficient and easily understandable for articulated 3D shape retrieval.

References

1. Agathos, A., Pratikakis, I., Papadakis, P., Perantonis, S., Azariadis, P., Sapidis, N.: Retrieval of 3D articulated objects using a graph-based representation. *The Visual Computer* 26, 1301–1319 (2010)
2. Au, O.K.-C., Tai, C.-L., Cohen-Or, D., Zheng, Y., Fu, H.: Electors voting for fast automatic shape correspondence. In: *Proc. Eurographics*, vol. 29 (2010)

3. Au, O.K.-C., Tai, C.-L., Chu, H.-K., Cohen-Or, D., Lee, T.-Y.: Skeleton extraction by mesh contraction. *ACM Transactions on Graphics* 27 (2008)
4. Bai, X., Latecki, L.J.: Path similarity skeleton graph matching. *IEEE Trans. Pattern Analysis and Machine Intelligence* 30, 1282–1292 (2008)
5. Belongie, S., Malik, J., Puzicha, J.: Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Analysis and Machine Intelligence* 24, 509–522 (2002)
6. Chen, D.-Y., Tian, X.-P., Shen, Y.-T., Ouhyoung, M.: On visual similarity based 3D model retrieval. *Computer Graphics Forum* 22, 223–232 (2003)
7. Cornea, N.D., Demirci, M.F., Silver, D., Shokoufandeh, A., Dickinson, S., Kantor, P.B.: 3D object retrieval using many-to-many matching of curve skeletons. In: *Proc. Int. Conf. Shape Modeling and Applications*, pp. 368–373 (2005)
8. Cornea, N.D., Silver, D., Yuan, X., Balasubramanian, R.: Computing hierarchical curve-skeletons of 3D objects. *The Visual Computer* 21, 945–955 (2005)
9. Cornea, N.D., Silver, D., Min, P.: Curve-skeleton properties, applications, and algorithms. *IEEE Trans. Visualization and Computer Graphics* 13, 530–548 (2007)
10. Lian, Z., Godil, A., Fabry, T., Furuya, T., Hermans, J., Ohbuchi, R., Shu, C., Smeets, D., Suetens, P., Vandermeulen, D., Wuhrer, S.: SHREC 2010 Track: Non-rigid 3D shape retrieval. In: *Proc. Eurographics Workshop on 3D Object Retrieval*, pp. 1–8 (2010)
11. Kazhdan, M., Chazelle, B., Dobkin, D., Finkelstein, A., Funkhouser, T.: A reflective symmetry descriptor. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) *ECCV 2002*. LNCS, vol. 2351, pp. 642–656. Springer, Heidelberg (2002)
12. Shilane, P., Min, P., Kazhdan, M., Funkhouser, T.: The Princeton shape benchmark. In: *Proc. Shape Modeling International*, pp. 167–178 (2004)
13. Shinagawa, Y., Kunii, T.L., Kergosien, Y.L.: Surface coding based on Morse theory. *IEEE Computer Graphics and Applications* 11, 66–78 (1991)
14. Siddiqi, K., Shokoufandeh, A., Dickinson, S., Zucker, S.: Shock graphs and shape matching. *International Journal of Computer Vision* 35, 13–32 (1999)
15. Siddiqi, K., Zhang, J., Macrini, D., Shokoufandeh, A., Bouix, S., Dickinson, S.: Retrieving articulated 3-D models using medial surfaces. *Machine Vision and Applications* 19, 261–275 (2008)
16. Sundar, H., Silver, D., Gagvani, N., Dickinson, S.: Skeleton based shape matching and retrieval. In: *Proc. Shape Modeling International*, pp. 130–139 (2003)
17. Wang, Y.-S., Lee, T.-Y.: Curve skeleton extraction using iterative least squares optimization. *IEEE Trans. Visualization and Computer Graphics* 14, 926–936 (2008)
18. Wu, H.-Y., Zha, H., Luo, T., Wang, X.-L., Ma, S.: Global and local isometry-invariant descriptor for 3D shape comparison and partial matching. In: *Proc. CVPR*, pp. 438–445 (2010)
19. <http://3d.csie.ntu.edu.tw/~dynamic/3DRetrieval/index.html>

The Number of Khalimsky-Continuous Functions between Two Points

Shiva Samieinia

Department of Mathematics, Stockholm University
shiva@math.su.se

Abstract. We determine the number of Khalimsky-continuous functions defined on an interval, having two fixed endpoints, and with values in \mathbb{Z} , in \mathbb{N} , or in a bounded interval. The number of Khalimsky-continuous functions with two points in their codomain gives an example of the Fibonacci sequence. A recurrence formula shall be presented to determine the number of Khalimsky-continuous functions with the values in a bounded interval. Using a generating function leads us to determine the number of increasing Khalimsky-continuous functions. Considering \mathbb{N} as a codomain of these functions yields a new example of the classical Fibonacci sequence.

Keywords: Digital geometry, Khalimsky plane, Khalimsky-continuous functions.

1 Introduction

Digital geometry has been called the geometry of the computer screen. After the advent of the computer and developing the technique in handling digital pictures through image processing tools, it became an important theoretical framework for image processing. In digital geometry we need to redefine geometric objects from Euclidean geometry such as line, curve, surface, and also establish some new criteria for the study of discrete planes. The book by Klette and Rosenfeld [10] provides a good survey of almost everything in this field. The lecture notes by Kiselman [8] contain concepts of digital geometry in simple mathematical formulations.

The subject is growing fast towards developing computer techniques and there are many recent publications in this area. Some objects have been of special interest and been studied a long time. Digital lines and hyperplanes are such objects. Digital lines were studied arithmetically by Reveillès [15] by means of double Diophantine inequalities. As a generalization of this, naive digital hyperplanes are obtained. Kiselman [7, 9] generalized Reveillès' definition of a digital hyperplane by allowing more freely strict and non-strict inequalities.

Digital lines were also studied geometrically. We mention the work by Rosenfeld [17] in which he defined the chord property, the work by Ronse [16] on the strong chord property, and Sharaiha [21] on the compact chord property.

The concept of runs was introduced in [17] and continued by Smeulders and Dorst [22], Stephenson [23] and Uscka-Wehlou [24].

A pioneering and fundamental way to study digital arcs is by using codes. Maloñ and Freeman [12] and Freeman [3] introduced the chain code as a technique for representing 8-connected arcs and lines. The chain code of an arc is a sequence the elements of which can be one of the values $0, \dots, 7$. The corresponding chain code of lines with slope $0 \leq \alpha \leq 1$ can contain only zeros and ones.

As combinatorial work on digital objects we can mention the work by Huxley and Zunić [5] on the number of different digital discs consisting of N points. The number of discrete segments with slope $0 \leq \alpha \leq 1$ and length L was studied by Berenstein and Lavine [2]. The number of digital straight lines on an $\mathbb{N} \times \mathbb{N}$ grid was determined by Koplowitz et al. [11]. Work on the number of digital straight line segments was done by Bédaride et al. [1] in order to determine the number of digital straight line segments of given length and height.

Digital geometry is providing a method to transform real objects to discrete ones, digitization. However, digital geometry does not consist just of digitizing Euclidean objects. In some cases we need to define a discrete object independently of the digitization process to provide a good model. A function $\mathbb{Z} \rightarrow \mathbb{Z}$ is not in general a good model for curves in digital geometry even though it can be a digitization of some real functions. As is the case for a function of real variables, it is convenient to require it to be continuous. To do that we need to define a topology on \mathbb{Z}^2 to define the concept of continuity using open sets and the inverse images of these. At the same time, we need a topology which provides a connectivity that makes all \mathbb{Z}^2 connected. Khalimsky topology is a suitable choice. It was introduced by Khalimsky [4], and developed towards problems in digital geometry by Khalimsky et al. [6]. After equipping the discrete plane \mathbb{Z}^2 with a topology, as we expect, we are able to speak about a continuous function. For more information in these subjects see Kiselman [8] and Melin [13] and [14].

Enumeration of Khalimsky-continuous functions was studied by Samieinia [18], [19] and [20]. We studied these functions when they have two points in their codomain, and it yields a new example of the classical Fibonacci sequence. For the case of three or four points in their codomain, some new sequences were presented (see [19]). We also determined the number of such functions on a given interval. In this case it turned out that these numbers are related to the Delannoy and Schröder arrays (see [20]).

From Euclidean geometry we know that there are just one line segment between two points, but in the digital plane this is not true. The other difference between the world of real numbers and the discrete world is the number of digital arcs between two points. In \mathbb{R}^2 we have infinitely many continuous functions $\mathbb{R} \rightarrow \mathbb{R}$ between two points, but we do not have the same situation for the Khalimsky-continuous functions $\mathbb{Z} \rightarrow \mathbb{Z}$. Although there are many discrete functions with two fixed endpoints, it is possible to count them.

In Section 2 we define the Khalimsky topology on \mathbb{Z}^2 and then we review the definition of Khalimsky-continuous function. In Section 3 we consider six

different cases for enumerating the Khalimsky-continuous functions. This classification depends on the codomain and on the fact whether the functions are required to be increasing or not. The first and the third case deal with the number of Khalimsky-continuous functions when the codomain is \mathbb{Z} or \mathbb{N} . These cases were studied in [20] by the author. These two cases give examples of the Delannoy and Schröder numbers. Considering increasing Khalimsky-continuous functions with codomain \mathbb{Z} and \mathbb{N} implies that the values for these two cases are exactly the same. The number of Khalimsky-continuous functions when the codomain contains only two elements gives an example of Fibonacci sequence. The fifth and sixth cases deal with the Khalimsky-continuous functions when they have a finite codomain possibly with more than two points. These numbers are given by a recurrence formula. The exact value for the increasing Khalimsky-continuous functions shall be determined by introducing a generating function.

2 The Khalimsky Topology and Khalimsky-Continuous Functions

We present the Khalimsky topology using a topological basis. For every even integer m , the set $\{m - 1, m, m + 1\}$ is open, and for every odd integer n , the singleton set $\{n\}$ is open. A basis is given by

$$\{\{2n + 1\}, \{2n - 1, 2n, 2n + 1\}; n \in \mathbb{Z}\}.$$

It follows that even points are closed. A digital interval $[a, b]_{\mathbb{Z}} = [a, b] \cap \mathbb{Z}$ with the subspace topology is called a *Khalimsky interval*, and a homeomorphic image of a Khalimsky interval into a topological space is called a *Khalimsky arc*. Figure 1 illustrates the Khalimsky line.

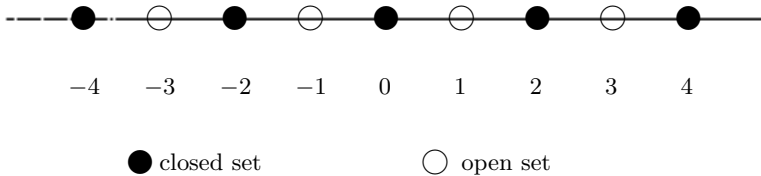


Fig. 1. The Khalimsky line

On the digital plane \mathbb{Z}^2 , the Khalimsky topology is given by the product topology. A point with both coordinates odd is open. If both coordinates are even, the point is closed. These types of points are called *pure*. Points with one even and one odd coordinate are neither open nor closed; these are called *mixed*. We can see easily that the mixed point $m = (m_1, m_2)$ is connected in the topological sense to its four neighbors,

$$(m_1 \pm 1, m_2) \text{ and } (m_1, m_2 \pm 1),$$

whereas the pure point $p = (p_1, p_2)$ is connected to all its 8-neighbors,

$$(p_1 \pm 1, p_2), (p_1, p_2 \pm 1), (p_1 + 1, p_2 \pm 1) \text{ and } (p_1 - 1, p_2 \pm 1).$$

More information on the Khalimsky plane and the Khalimsky topology can be found in [8].

When we equip \mathbb{Z} with the Khalimsky topology, we may speak of continuous functions $\mathbb{Z} \rightarrow \mathbb{Z}$. It is easy to see that a continuous function f is Lipschitz with constant 1. This is however not sufficient for continuity. It is not hard to prove that $f: \mathbb{Z} \rightarrow \mathbb{Z}$ is continuous if and only if (i) f is Lip-1 and (ii) for every $x \not\equiv f(x) \pmod{2}$, $f(x \pm 1) = f(x)$. For more information see Melin [13] and [14].

Also, we observe that the following functions are continuous:

- (1) $\mathbb{Z} \ni x \mapsto a \in \mathbb{Z}$, where a is constant;
- (2) $\mathbb{Z} \ni x \mapsto \pm x + c \in \mathbb{Z}$, where c is an even constant;
- (3) $\max(f, g)$ and $\min(f, g)$ if f and g are continuous.

Actually every continuous function on a bounded Khalimsky interval can be obtained by a finite succession of the rules (1), (2), (3); (see Kiselman [8]). The Khalimsky plane is illustrated in Figure 2.

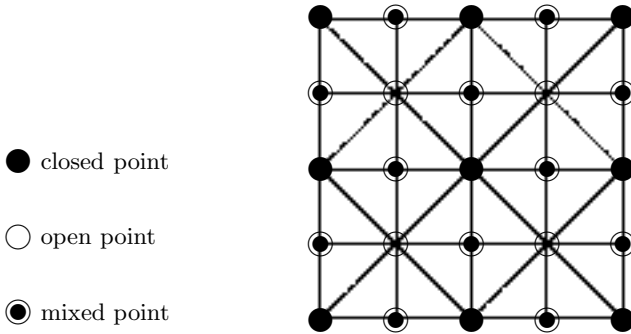


Fig. 2. The Khalimsky plane

3 Khalimsky-Continuous Functions with Two Fixed Endpoints

Here we are interested in finding the number of Khalimsky-continuous functions with two fixed endpoints. We define six different cases as follows:

- Definition 1.**
- 1. Let A_n^s denote the number of Khalimsky-continuous functions $f: [0, n]_{\mathbb{Z}} \rightarrow \mathbb{Z}$ such that $f(0) = 0$ and $f(n) = s$.
 - 2. Let a_n^s denote the number of increasing Khalimsky-continuous functions $f: [0, n]_{\mathbb{Z}} \rightarrow \mathbb{Z}$ such that $f(0) = 0$ and $f(n) = s$.
 - 3. Let B_n^s denote the number of Khalimsky-continuous functions $f: [0, n]_{\mathbb{Z}} \rightarrow \mathbb{N}$ such that $f(0) = 0$ and $f(n) = s$.

4. Let b_n^s denote the number of increasing Khalimsky-continuous functions $f: [0, n]_{\mathbb{Z}} \rightarrow \mathbb{N}$ such that $f(0) = 0$ and $f(n) = s$.
5. Let $C_n^{s,t}$ denote the number of Khalimsky-continuous functions $f: [0, n]_{\mathbb{Z}} \rightarrow [0, s]_{\mathbb{Z}}$ such that $f(0) = 0$ and $f(n) = t$.
6. Let $c_n^{s,t}$ denote the number of increasing Khalimsky-continuous functions $f: [0, n]_{\mathbb{Z}} \rightarrow [0, s]_{\mathbb{Z}}$ such that $f(0) = 0$ and $f(n) = t$.

The first and the third cases were studied in Samieinia [20].

Theorem 1 (Samieinia [20]). Let A_n^s , $|s| \leq n$, be the number of Khalimsky-continuous functions $f: [0, n]_{\mathbb{Z}} \rightarrow \mathbb{Z}$ such that $f(0) = 0$ and $f(n) = s$, and $d_{i,j}$ be the Delannoy numbers. Then we have that $A_n^s = d_{i,j}$ for $i = \frac{1}{2}(n + s)$ and $j = \frac{1}{2}(n - s)$ where $n + s \in 2\mathbb{Z}$, and $A_n^s = A_{n-1}^s$ for $n + s$ odd.

Theorem 2 (Samieinia [20]). Let B_n^s be the number of Khalimsky-continuous functions $f: [0, n]_{\mathbb{Z}} \rightarrow \mathbb{N}$ such that $f(0) = 0$ and $f(n) = s$ for $s \in \mathbb{N}$ and $s \leq n$, and $r_{i,j}$ be the Schröder numbers. Then we have $B_n^s = r_{i,j}$ for $i = \frac{1}{2}(n + s)$ and $j = \frac{1}{2}(n - s)$, where $n + s \in 2\mathbb{N}$.

Considering increasing Khalimsky-continuous functions with codomain \mathbb{N} yields an example of the Fibonacci sequence.

Theorem 3. Let b_n be the number of increasing Khalimsky-continuous functions $f: [0, n] \rightarrow \mathbb{N}$ such that $f(0) = 0$. Then $b_n = F_{n+2}$, $n \in \mathbb{N}$, where $(F_n)_0^\infty$ is the Fibonacci sequence, defined by $F_0 = 0$, $F_1 = 1$, $F_n = F_{n-1} + F_{n-2}$, $n \geq 2$.

Proof. Let b_n^s be the number of increasing Khalimsky-continuous functions $f: [0, n] \rightarrow \mathbb{N}$ such that $f(0) = 0$ and $f(n) = s$. Using the properties of the Khalimsky topology we have that

$$b_n = b_n^0 + b_n^1 + \dots + b_n^n. \tag{1}$$

By the properties of the Khalimsky topology for n, s of the same parity we have

$$b_n^s = b_{n-1}^{s-1} + b_{n-1}^s, \tag{2}$$

while for n, s of opposite parity,

$$b_n^s = b_{n-1}^s. \tag{3}$$

Inserting equations (2) and (3) into (1) will give the result.

In the next table we can see the values of b_n^s and their sum b_n .

$s \setminus n$	0	1	2	3	4	5	6	7	8	9
0	1	1	1	1	1	1	1	1	1	1
1	0	1	1	2	2	3	3	4	4	5
2	0	0	1	1	3	3	6	6	10	10
3	0	0	0	1	1	4	4	10	10	20
4	0	0	0	0	1	1	5	5	15	15
5	0	0	0	0	0	1	1	6	6	21
6	0	0	0	0	0	0	1	1	7	7
7	0	0	0	0	0	0	0	1	1	8
8	0	0	0	0	0	0	0	0	1	1
9	0	0	0	0	0	0	0	0	0	1
b_n	1	2	3	5	8	13	21	34	55	89

To study the number $C_n^{s,t}$ defined in [1], we first consider the Khalimsky-continuous functions when they have two points in their codomain.

Proposition 1. *Let $C_n^{1,t}$ be the number of Khalimsky-continuous functions $f: [0, n]_{\mathbb{Z}} \rightarrow [0, 1]_{\mathbb{Z}}$ such that $f(0) = 0$ and $f(n) = t$. Then for $k \geq 1$, $C_{2k}^{1,1} = C_{2k-1}^{1,1} = F_{2k}$ and $C_{2k+1}^{1,0} = C_{2k}^{1,0} = F_{2k+1}$ where F_n is the Fibonacci number.*

Proof. By the properties of the Khalimsky topology we have

$$\begin{aligned}
 C_{2k+1}^{1,0} &= C_{2k}^{1,0}, & k \geq 1, \\
 C_{2k}^{1,0} &= C_{2k-1}^{1,0} + C_{2k-1}^{1,1}, & k \geq 1,
 \end{aligned}
 \tag{4}$$

and

$$\begin{aligned}
 C_{2k}^{1,1} &= C_{2k-1}^{1,1}, & k \geq 1, \\
 C_{2k+1}^{1,1} &= C_{2k}^{1,0} + C_{2k}^{1,1}, & k \geq 1.
 \end{aligned}
 \tag{5}$$

It is clear that $C_1^{1,1} = C_1^{1,0} = 1$. By using equations (4) and (5) we have

$$C_{2k}^{1,1} = C_{2k-1}^{1,1} = C_{2k-2}^{1,0} + C_{2k-2}^{1,1},$$

and

$$C_{2k}^{1,0} = C_{2k-1}^{1,0} + C_{2k-1}^{1,1}.$$

Now using an induction can give the result.

In the next table we can see the number of $C_n^{1,0}$ and $C_n^{1,1}$ for $1 \leq n \leq 11$.

n	1	2	3	4	5	6	7	8	9	10	11
$C_n^{1,0}$	1	2	2	5	5	13	13	34	34	89	89
$C_n^{1,1}$	1	1	3	3	8	8	21	21	55	55	144

As we have seen the number of Khalimsky continuous functions with two points in their codomain presented an example of the known Fibonacci sequence. In the following theorem we consider the Khalimsky-continuous functions with bounded codomain possibly with more than two points.

Proposition 2. *Let $C_n^{s,t}$ be the number of Khalimsky-continuous functions $f: [0, n]_{\mathbb{Z}} \rightarrow [0, s]_{\mathbb{Z}}$ such that $f(0) = 0, f(n) = t$ for $0 \leq t \leq s$. Then for n, s of the same parity we have*

$$C_n^{s,s} = C_{n-1}^{s,s-1} + C_{n-3}^{s,s-1} + \dots + C_{s+1}^{s,s-1} + C_{s-1}^{s,s-1}, \quad 1 \leq s \leq n, \quad (6)$$

while for n, s of opposite parity,

$$C_n^{s,s} = C_{n-1}^{s,s}, \quad 1 \leq s \leq n. \quad (7)$$

Proof. Suppose that n and s have the same parity. From the Lipschitz property it follows that

$$C_n^{s,s} = C_{n-1}^{s,s} + C_{n-1}^{s,s-1}. \quad (8)$$

Since n and s have the same parity, from the properties of the Khalimsky topology

$$C_{n-1}^{s,s} = C_{n-2}^{s,s}, \quad (9)$$

so

$$C_{n-2}^{s,s} = C_{n-3}^{s,s} + C_{n-3}^{s,s-1}. \quad (10)$$

Hence, by using in turn (8), (9) and (10), we have

$$C_n^{s,s} = C_{n-1}^{s,s-1} + C_{n-3}^{s,s-1} + C_{n-3}^{s,s}, \quad n > 3. \quad (11)$$

In the same way we can continue for $C_{n-3}^{s,s}$ and so on, until we get $C_s^{s,s}$. Since $C_s^{s,s} = C_{s-1}^{s,s-1} = 1$, we get the result.

The next table shows the values of $C_n^{5,t}$, which are the number of Khalimsky-continuous functions $f: [0, n]_{\mathbb{Z}} \rightarrow [0, 5]_{\mathbb{Z}}, f(0) = 0, f(n) = t$ for $0 \leq t \leq 5$ and $0 \leq n \leq 9$.

$s \setminus n$	0	1	2	3	4	5	6	7	8	9
0	1	1	2	2	6	6	22	22	90	90
1	0	1	1	4	4	16	16	68	68	304
2	0	0	1	1	6	6	30	30	146	146
3	0	0	0	1	1	8	8	48	48	263
4	0	0	0	0	1	1	10	10	69	69
5	0	0	0	0	0	1	1	11	11	80

Considering the increasing Khalimsky-continuous functions with a bounded codomain (possibly with more than two points) implies the same recurrence formula as (6).

Proposition 3. Let $c_n^{s,t}$ be the number of increasing Khalimsky-continuous functions $f: [0, n]_{\mathbb{Z}} \rightarrow [0, s]_{\mathbb{Z}}$ such that $f(0) = 0, f(n) = t$ for $0 \leq t \leq s$. Then for n, s of the same parity we have

$$c_n^{s,s} = c_{n-1}^{s,s-1} + c_{n-3}^{s,s-1} + \dots + c_{s+1}^{s,s-1} + c_{s-1}^{s,s-1}, \quad 1 \leq s \leq n, \tag{12}$$

while for n, s of opposite parity,

$$c_n^{s,s} = c_{n-1}^{s,s}, \quad 1 \leq s \leq n. \tag{13}$$

Proof. The proof can be done in the same way as Proposition 2.

Now we shall define a generating function g_n with $c_n^{s,s}$ as its coefficients. Using this generating function we determine the number of increasing Khalimsky-continuous functions $c_{n+2i}^{n,n}$ for a fixed natural number $n \geq 2$ and for $i \in \mathbb{N}$. By the Khalimsky topology, we can see easily that $c_{n+2i+1}^{n,n} = c_{n+2i}^{n,n}$. Thus we just need to compute one of them. Let

$$g_n(x) = c_n^{n,n}x^0 + c_{n+2}^{n,n}x^2 + \dots = \sum_{i=0}^{\infty} c_{n+2i}^{n,n}x^{2i}. \tag{14}$$

By Proposition 3 and equation (14) we have

$$g_n(x) = c_{n-1}^{n,n-1}(x^0 + x^2 + \dots + x^{2i} + \dots) + c_{n+1}^{n,n-1}(x^2 + x^4 + \dots + x^{2i} + \dots) + \dots \tag{15}$$

We can get easily the following relation for the generator function g_n .

Proposition 4. Let $n \geq 2$ and denote by g_n the generating function of the sequence $(c_{n+2i}^{n,n})_{i=0}^{\infty}$. It satisfies

$$(1 - x^2)g_n(x) = g_{n-1}(x). \tag{16}$$

Proof. By equation (15),

$$x^2 g_n(x) = c_{n-1}^{n,n-1}(x^2 + x^4 + \dots + x^{2i+2} + \dots) + \dots \tag{17}$$

Thus by (17), (15) and (14);

$$g_n(x) - x^2 g_n(x) = x^0 c_{n-1}^{n,n-1} + x^2 c_{n+1}^{n,n-1} + \dots + x^m c_{n+m-1}^{n,n-1} + \dots = g_{n-1}(x).$$

If we use $n - 1$ times the equation (12), we get

$$g_1(x) = (1 - x^2)^{n-1} g_n(x) \text{ for } n \geq 2. \tag{18}$$

Using equation (18) leads us to find the number of increasing Khalimsky-continuous functions which is stated as follows:

Theorem 4. Let $c_{n+2j}^{n,t}$ be the number of increasing Khalimsky-continuous functions $f: [0, n + 2j]_{\mathbb{Z}} \rightarrow [0, n]_{\mathbb{Z}}$ such that $f(0) = 0$, $f(n + 2j) = t$ for $0 \leq t \leq n$ and $j \in \mathbb{N}$. Then

$$c_{n+2k}^{n,n} = \sum_{\substack{i,j \in \mathbb{N} \\ i+j=k}} \binom{n+i-2}{i} c_{1+2j}^{n,1},$$

which is, more precisely,

$$\binom{n+k-2}{k} + 2\binom{n+k-3}{k-1} + \dots + (k+1)\binom{n-2}{0}. \tag{19}$$

Proof. By using (18) and (14);

$$\begin{aligned} g_n(x) &= (1 - x^2)^{-(n-1)} g_1(x) \\ &= \left(\sum_{i=0}^{\infty} \binom{n+i-2}{i} x^{2i} \right) \left(\sum_{j=0}^{\infty} c_{1+2j}^{n,1} x^{2j} \right). \end{aligned} \tag{20}$$

Now a simple calculation and comparing the coefficient of x^{2k} of the right and the left sides of (20) together with the fact that $c_{1+2j}^{n,1} = j + 1$ for $j \in \mathbb{N}$ give the result.

The next table shows the values of $c_n^{5,t}$, which are the number of increasing Khalimsky-continuous functions $f: [0, n]_{\mathbb{Z}} \rightarrow [0, 5]_{\mathbb{Z}}$ such that $f(0) = 0$ and $f(n) = t$ for $0 \leq n \leq 9$ and $0 \leq t \leq 5$.

$s \setminus n$	0	1	2	3	4	5	6	7	8	9
0	1	1	1	1	1	1	1	1	1	1
1	0	1	1	2	2	3	3	4	4	5
2	0	0	1	1	3	3	6	6	10	10
3	0	0	0	1	1	4	4	10	10	20
4	0	0	0	0	1	1	5	5	15	15
5	0	0	0	0	0	1	1	6	6	21

It is easy to see that $a_n^s = b_n^s = c_n^{s,s}$. Therefore, we can determine a_n^s and b_n^s by knowing the values of $c_n^{s,t}$ which was determined in equation (19). We notice that although we have the same result of Theorem 4 for the values of $C_n^{s,t}$, the generator function g_n cannot be used to determine those values.

4 Conclusion

There are contrasts between the objects and criteria in the real plane as compared with the discrete plane. From Euclid’s first Postulate, we may conclude that there is a line segment between two points. He did not say that we have just one line, but we may conclude that this is so. However, there are usually

more than one digital straight line segment between two points in the digital plane. The other contrast between the Euclidean and the discrete planes are exhibited by digital curves. There are as many smooth curves as we like between two points in the Euclidean plane, but not in the discrete plane. The interesting fact for the discrete case is that we may count them.

In this paper we studied six different cases for the number of Khalimsky-continuous functions when they have two fixed endpoints. Two of these six cases were studied in a previous work by the author. We gave a recurrence formula to determine the number of Khalimsky-continuous functions not exiting from a rectangle and between two points. The number of increasing Khalimsky-continuous functions with two fixed endpoints and with values in a rectangle was also studied by using a generating function. The exact value for the number of increasing Khalimsky-continuous functions was established. The sum of the values of those numbers when the functions are in a rectangle with the same length and different heights gave an example of the classical Fibonacci sequence. These are the number of increasing Khalimsky-continuous functions with just one fixed endpoint (the origin) and \mathbb{N} as codomain.

Acknowledgment

I wish to thank Christer Kiselman and Svante Linusson for their comments on earlier versions of this manuscript.

References

1. Bédaride, N., Domenjoud, E., Jamet, D., Rémy, J.-L.: On the number of balanced words of given length and height over a two-letter alphabet. *Discrete Math. Theor. Comput. Sci.* 12(3), 41–62 (2010)
2. Berenstein, C.A., Lavine, D.: On the number of digital straight line segments. *P.A.M.I.* 10(6), 880–887 (1988)
3. Freeman, H.: Boundary encoding and processing. In: Lipkin, B.S., Rosenfeld, A. (eds.) *Picture Processing and Psychopictorics*, pp. 241–266 (1970)
4. Halimskii, E.D.: On topologies of generalized segments. *Soviet Math. Dokl.* 10, 1508–1511 (1969)
5. Huxley, M.N., Žunić, J.: The number of N -point digital discs. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29(1), 159–161 (2007)
6. Khalimsky, E.D., Kopperman, R.D., Meyer, P.R.: Computer graphics and connected topologies on finite ordered sets. *Topology Appl.* 36(1), 1–17 (1990)
7. Kiselman, C.O.: Convex functions on discrete sets. In: Klette, R., Žunić, J. (eds.) *IWCIA 2004. LNCS*, vol. 3322, pp. 443–457. Springer, Heidelberg (2004)
8. Kiselman, C.O.: *Digital geometry and mathematical morphology*. Lecture notes, Uppsala University (2004)
9. Kiselman, C.O.: Characterizing digital straightness by means of difference operators. In: Luengo, C., Gavrilovic, M. (eds.) *Proceedings SSBA 2010*, pp. 15–18 (2010)
10. Klette, R., Rosenfeld, A.: *Digital Geometry – Geometric Methods for Digital Picture Analysis*. Morgan Kaufmann, San Francisco (2004)

11. Koplowitz, J., Lindenbaum, M., Bruckstein, A.: The number of digital straight lines on an $\mathbb{N} \times \mathbb{N}$ grid. *IEEE Transactions on Information Theory* 36(1), 192–197 (1990)
12. Maloň, S., Freeman, H.: On the encoding of arbitrary geometric configurations. *IRE Trans. EC-10*, 260–268 (1961)
13. Melin, E.: Extension of continuous functions in digital spaces with the Khalimsky topology. *Topology Appl.* 153, 52–65 (2005)
14. Melin, E.: *Digital Geometry and Khalimsky Spaces*. Ph.D. Thesis, Uppsala Dissertations in Mathematics 54. Uppsala University (2008)
15. Reveillès, J.-P.: *Géométrie discrète, calcul en nombres entiers et algorithmique*. Ph.D. Thesis, Université Louis Pasteur, Strasbourg (1991)
16. Ronse, C.: A strong chord property for 4-connected convex digital sets. *Computer vision, Graphics and Image Processing* 35, 259–269 (1986)
17. Rosenfeld, A.: Digital straight line segments. *IEEE Transactions on Computers* C-32(12), 1264–1269 (1974)
18. Samieinia, S.: Chord Properties of digital straight line segments. *Math. Scand.* 106, 169–195 (2010)
19. Samieinia, S.: The number of Khalimsky-continuous functions on intervals. *Rocky Mountain J. Math.* 40(5), 1667–1687 (2010)
20. Samieinia, S.: The number of continuous curves in digital geometry. *Port. Math.* 67(1), 75–89 (2010)
21. Sharaiha, Y.M., Garat, P.: A compact chord property for digital arcs. *Pattern recognition* 26(5), 799–803 (1993)
22. Smeulders, A.W.M., Dorst, L.: Decomposition of discrete curves into piecewise straight segments in linear time. *Contemporary Math.* 119, 169–195 (1991)
23. Stephenson, P.D.: *The structure of the digitised line: with applications to line drawing and ray tracing in computer graphics*. Ph.D. Thesis, James Cook University of North Queensland, Department of Computer Science (1998)
24. Uscka-Wehlou, H.: *Digital Lines, Sturmian Words, and Continued Fractions*. Ph.D. Thesis, Uppsala Dissertations in Mathematics 65. Uppsala University (2009)

Cup Products on Polyhedral Approximations of 3D Digital Images

Rocio Gonzalez-Diaz¹, Javier Lamar², and Ronald Umble³

¹ Dept. of Applied Math (I), School of Computer Engineering, University of Seville, Campus Reina Mercedes, C.P. 41012, Seville, Spain

rogodi@us.es

² Pattern Recognition Department, Advanced Technologies Application Center, 7th Avenue #21812 218 and 222, Siboney, Playa, C.P. 12200, Havana City, Cuba

jlamar@cenatav.co.cu

³ Department of Mathematics, Millersville University of Pennsylvania, P.O. Box 1002 Millersville, PA 17551-0302, Pennsylvania, USA

ron.umble@millersville.edu

Abstract. Let I be a 3D digital image, and let $Q(I)$ be the associated cubical complex. In this paper we show how to simplify the combinatorial structure of $Q(I)$ and obtain a homeomorphic cellular complex $P(I)$ with fewer cells. We introduce formulas for a diagonal approximation on a general polygon and use it to compute cup products on the cohomology $H^*(P(I))$. The cup product encodes important geometrical information not captured by the cohomology groups. Consequently, the ring structure of $H^*(P(I))$ is a finer topological invariant. The algorithm proposed here can be applied to compute cup products on any polyhedral approximation of an object embedded in 3-space.

Keywords: Cellular complex, cohomology, cup product, diagonal approximation, digital image, polyhedron.

1 Introduction

Throughout this paper, coefficients lie in the field \mathbb{Z}_2 . Let X be a cellular complex embedded in 3-dimensional space and constructed by gluing 3-dimensional polyhedra together along common faces (see [4]). At a most basic level, the connected components, homotopy classes of non-contractible loops, and boundaries of tunnels in X generate the cellular cohomology $H^*(X)$. At the next level, certain relationships among the generators are encoded by the cup product, which endows $H^*(X)$ with a graded commutative ring structure. Indeed, the discriminating information encoded by the cup product improves our capability to distinguish between 3D images. For example, $H^*(S^1 \vee S^1 \vee S^2)$ and $H^*(S^1 \times S^1)$ are isomorphic as vector spaces but not as rings since cup products vanish in the wedge but not in the product. Thus $S^1 \vee S^1 \vee S^2$ and $S^1 \times S^1$ have quite different topological properties.

To date, the cup product has seen limited application to problems in 3D image processing. In [10,11], Gonzalez-Diaz and Real used their 14-adjacency algorithm

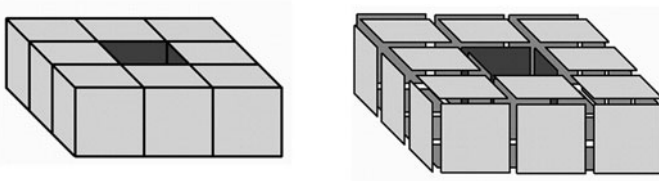


Fig. 1. Left: A digital image $I = (\mathbb{Z}^3, 26, 6, B)$; the set B consists of 8 unit cubes (voxels). Right: The quadrangles of $\partial Q(I)$.

and the standard formulation in [17] to compute cup products on the simplicial complex $K(I)$ associated with a given digital image I . More recently, Gonzalez-Diaz, Jimenez and Medrano introduced a method for computing cup products on cubical approximations $Q(I)$. Their cup products are computed directly from the cubical complex, and no additional subdivisions are necessary [8,9]. For a geometrical interpretation of cohomology in the context of digital images, we refer the reader to [5,6,15].

In [14], Kravatz computed cup products on a general 2-dimensional polygon in terms of a combinatorial *diagonal approximation*, which assumes a particular ordering of the vertices. In this paper, we introduce a more general formula for computing cup products, which is independent of the ordering of vertices and computationally effective.

A problem that frequently arises in 3D image processing is to efficiently encode the boundary surface of a given digital object as a set of voxels. The most popular approach to this problem uses a triangulation. While triangles are combinatorially simple, and visualization of triangulated surfaces is supported by existing hardware and software, the number of triangles required is often large and the computational analysis correspondingly slow. It is desirable, therefore, to seek more computationally economical combinatorial approximations. Adjacent coplanar triangles in a triangulation, for example, can be merged into more general polygons and become faces of more general but combinatorially simpler polyhedra. The payoff from combinatorial simplicity is improved computational efficiency.

Approximating 3D objects with polyhedral complexes is a well-studied problem in the field of Computational Geometry (for example, see [12,3]). An algorithm for constructing polyhedral approximations in certain special cases was given by Kovalevsky and Schulz in [13,19]. Their algorithm generates the convex hull of a given object then modifies the convex hull by recursively generating convex hulls of either subsets of the given voxel set or subsets of the background voxels. The result of this method is a polyhedron that separates object voxels from background voxels.

The computational methods introduced in this paper can be effectively applied to any polyhedral approximation of a 3D object. Indeed, one maximizes computational efficiency by approximating a given 3D object with a polyhedral

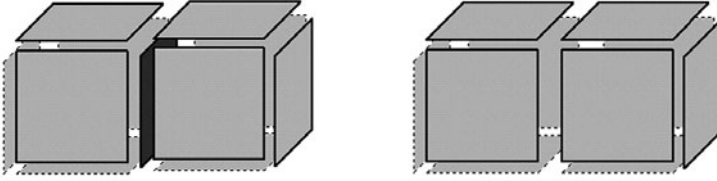


Fig. 2. Left: Quadrangles in the boundary of cubes c and σ sharing a square σ' (in bold). Right: Quadrangles in $\partial(c) := \partial(c + \sigma)$, the boundary of the cell c after removing σ' .

complex containing a minimal number of cells. To the extent that this is our long-term objective, we take a first step in this direction here.

The paper is organized as follows: In Section 2 we introduce a simplification procedure, which produces a cellular complex $P(I)$ homeomorphic to $Q(I)$ with significantly fewer cells. In Section 3 we define a diagonal approximation on a general polygon and use it to compute the cohomology ring of $P(I)$. Conclusions and some ideas for future work are discussed in Section 4.

2 3D Digital Pictures and Cellular Complexes

Let I be a 3D digital image and let $Q(I)$ be an associated cubical complex. In this section we introduce a simplification procedure, which produces a cellular complex $P(I)$ homeomorphic to $Q(I)$ with significantly fewer cells.

Intuitively, a *cellular decomposition* of a 3D space X embedded in \mathbb{R}^3 is a representation of X as a finite union of vertices (0-cells), edges (1-cells), polygons (2-cells), and polyhedra (3-cells), which have been glued together in such a way that the non-empty intersection of two cells is a cell. A k -cell is also referred to as a k -face. A *cellular complex* is a 3D space X embedded in \mathbb{R}^3 together with a cellular decomposition. For a precise definition of a cellular complex, which is more subtle than one might expect, see 4.

A *cubical complex* Q is a cellular complex whose 2-cells are squares (or quadrangles) and whose 3-cells are cubes. Note that if a cube is in Q , its bounding quadrangles are in Q ; if a quadrangle is in Q , its bounding edges are in Q ; and if an edge is in Q , its endpoints are in Q .

Consider a 3D binary digital picture $I = (\mathbb{Z}^3, 26, 6, B)$, where \mathbb{Z}^3 is the underlying grid and B (the foreground) is a finite set of points of the grid fixing the 26-adjacency for the points of B and the 6-adjacency for the points of $\mathbb{Z}^3 \setminus B$ (the background). The cells of $Q(I)$ are unit cubes centered at the points of B with faces parallel to the coordinate planes (called the *voxels* of I), together with their quadrangles, edges, and vertices.

Let K be a cellular complex. An i -cell $\sigma' \in K$ is a *facet* of a cell $\sigma \in K$ if σ is an $(i + 1)$ -cell and σ' is a face of σ . A *maximal* cell of K is not a facet of any cell of K . The boundary of K , denoted by ∂K , is the subcomplex of K consisting of all cells that are facets of exactly one (maximal) cell, and their faces. Note that

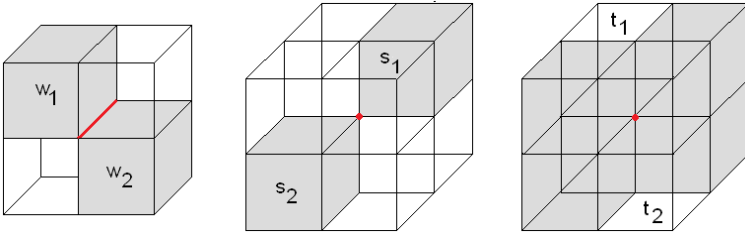


Fig. 3. Critical configurations (i), (ii) and (iii) (modulo reflections and rotations)

the maximal cells of $\partial Q(I)$ are all the quadrangles of $Q(I)$ shared by a voxel of B and a voxel of $\mathbb{Z}^3 \setminus B$ (see Figure 1).

Following the exposition in [8,9], given a digital image I and its associated cubical complex $Q(I)$, we apply a face-reduction technique to reduce the number of cells in $Q(I) \setminus \partial Q(I)$ and obtain a cellular complex $K(I)$ homeomorphic to $Q(I)$ whose maximal cells are the quadrangles of $\partial Q(I)$ (see Figure 2 and Algorithm 1). Then, $\partial K(I) = \partial Q(I)$.

```

INPUT: A cubical complex  $Q(I)$  associated to a 3D digital image  $I$ .
Initially,  $K(I) := Q(I)$ .
While there exists a cell  $\sigma' \in Q(I) \setminus \partial Q(I)$  do
  If  $\sigma'$  is a facet of exactly two cells  $c, \sigma \in Q(I)$  do
    remove  $\sigma$  and  $\sigma'$  from the current  $K(I)$ ;
    redefine  $c$  as  $c \cup \sigma$ .
  If  $\sigma'$  is a facet of exactly one cell  $\sigma \in Q(I)$  do
    remove  $\sigma$  and  $\sigma'$  from the current  $K(I)$ .
OUTPUT: the cellular complex  $K(I)$ .
    
```

Algorithm 1. Face-Reduction Process

Next, we perform a simplification process in $\partial K(I)$ to produce a cellular complex $P(I)$ homeomorphic to $K(I)$ such that the maximal cells of $\partial P(I)$ are polygons. But first, we need a definition.

Definition 1. A vertex $v \in \partial K(I)$ is **critical** if one of the following situations occurs:

- (i) v is a face of some edge e shared by four cubes, exactly two of which intersect along e and lie in $Q(I)$ (see cubes w_1 and w_2 in Figure 3).
- (ii) v is shared by eight cubes, exactly two of which are corner-adjacent and contained in $Q(I)$ (see cubes s_1 and s_2 in Figure 3).
- (iii) v is shared by eight cubes, exactly two of which are corner-adjacent and not contained in $Q(I)$ (cubes t_1 and t_2 in Figure 3).

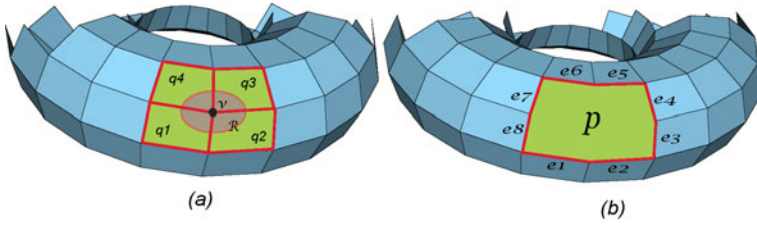


Fig. 4. (a) $N_v \leftarrow \{q_1, q_2, q_3, q_4\}$, (b) facets of $p \leftarrow \{e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8\}$

It is a well-known fact that a non-critical vertex of $\partial K(I)$ lies in a neighborhood of $\partial K(I)$ homeomorphic to \mathbb{R}^2 (see [16]).

Algorithm 2 processes the non-critical vertices of $\partial K(I)$ to obtain the cellular complex $P(I)$. Initially, $P(I) = K(I)$. For a vertex $v \in \partial P(I)$, let N_v be the set of 2-cells $q \in P(I)$ incident to the vertex v . If N_v defines a region R_v homeomorphic to a disc, then N_v is replaced by a new 2-cell p in $P(I)$, which is the union of the cells of N_v . The edges of $\partial P(I)$ incident to v and the vertex v are removed from $P(I)$ (see Figure 4). Observe that the maximal cells of the final cellular complex $\partial P(I)$ are polygons and $\partial P(I)$ has fewer cells than $\partial K(I)$. We can set some terminating conditions. For example: (1) terminate when the number of edges of the polygons in $\partial P(I)$ reach some specified maximum; or (2) terminate after merging the set N_v of coplanar 2-cells of $\partial P(I)$ (this preserves the geometry but removes fewer cells). An example of the differences that arise from these different terminating conditions is demonstrated in Example 1.

Observe that Algorithm 2 uses the ordering on the set of non-critical vertices $V \subset \partial K(I)$ to select the next non-critical vertex. To the best of our knowledge, this is the first algorithm to appear that produces a cellular complex with polygonal maximal cells by removing non-critical vertices.

Example 1. Let μI be a μ MRI of a trabecular bone of size: $85 \times 85 \times 10$ voxels (see Figure 5 in which μI is given by a sequence of 10 2D digital images of size

```

INPUT: The output of Algorithm 1: the cellular complex  $K(I)$ .
Initially,  $P(I) := K(I)$ ;
       $V :=$  ordered set of non-critical vertices of  $\partial K(I)$ .
While  $\exists v \in V$  such that  $R_v$  is homeomorphic to a disc do
  remove  $v$  from  $P(I)$  and  $V$ ;
  remove the edges incident to  $v$  from  $P(I)$ ;
  remove the 2-cells of  $N_v$  from  $P(I)$ ;
  add a new 2-cell  $p$  to  $P(I)$  which is the union of the cells of  $N_v$ .
OUTPUT: The cellular complex  $P(I)$ .

```

Algorithm 2. Algorithm to obtain the cellular complex $P(I)$

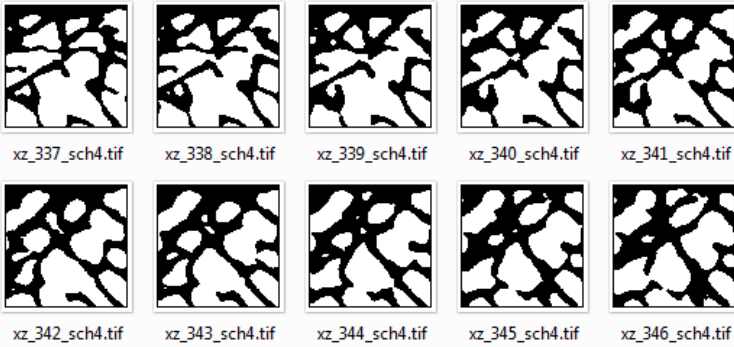


Fig. 5. A μ MRI of a trabecular bone

85×85). The number of quadrangles in $\partial Q(\mu I)$ is 20956 (see Figure 6). After applying Algorithm 1 to obtain the cellular complex $K(\mu I)$, we apply Algorithm 2 to $K(\mu I)$ with the terminating condition 1 (the number of the edges of the polygons in $\partial P(\mu I)$ is smaller or equal to 10). Then, the number of polygons of $\partial P(\mu I)$ is 1567 (see Figure 7a). If we only consider the set of coplanar polygons N_v , then the number of polygons of $\partial P(\mu I)$ after applying Algorithm 2 is 9321 (see Figure 7b).

3 Computing the Cohomology Ring of $P(I)$

Traditionally, one computes cup products in simplicial or cubical complex using the standard formulas in [17,20]. In this section, we give a procedure for computing cup products on $P(I)$ (the output of Algorithm 2), which avoids triangulation by defining explicit formulas for diagonal approximations on polygons.

We begin with a review of some standard definitions from Algebraic Topology (for details see [17]). Given a graded set $S = \{S_q\}_q$, the q -chains of S , which are finite formal sums of elements of S_q , define an additive abelian group structure on S_q . These groups, called q -chain groups, are denoted by $C_q(S)$. The collection of all chain groups associated with S is denoted by $C_*(S) = \{C_q(S)\}_q$

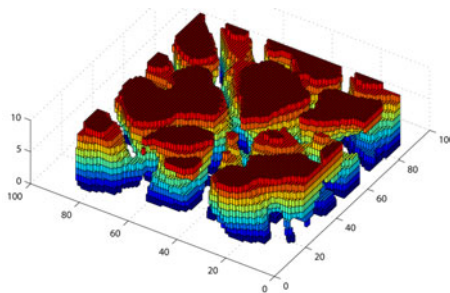


Fig. 6. The cubical complex $\partial Q(\mu I)$

and is referred to as the *chain group of S*. A *chain complex* $(C_*(S), \partial)$ is a chain group $C_*(S)$ together with a square zero homomorphism $\partial = \{\partial_q : C_q(S) \rightarrow C_{q-1}(S)\}_q$, called the *boundary operator*. For example, consider a triangle $\langle v_i, v_j, v_k \rangle$ with vertices $v_i < v_j < v_k$. The boundary of the triangle is the formal sum of its edges, that is, $\partial_2(\langle v_i, v_j, v_k \rangle) = \langle v_i, v_j \rangle + \langle v_j, v_k \rangle + \langle v_i, v_k \rangle$. Note that any chain group $C_*(S)$ together with the zero boundary map $\partial \equiv 0$ is a chain complex.

The chain complex associated with $P(I)$ (the output of Algorithm 2) is the collection $(C_*(P(I)), \partial) = \{C_q(P(I)), \partial_q\}_q$ where:

- each $C_q(P(I))$ is the chain group generated by the q -cells of $P(I)$,
- the boundary $\partial_q : C_q(P(I)) \rightarrow C_{q-1}(P(I))$ evaluated on a q -cell of $P(I)$ is the formal sum of its facets, and
- the boundary of a general q -chain is defined by linearly extending ∂ .

Given a chain complex $(C_*(S), \partial)$, a q -chain $\sigma \in C_q(S)$ is called a *q-cycle* if $\partial_q(\sigma) = 0$. If $\sigma = \partial_{q+1}(\mu)$ for some $(q+1)$ -chain μ then σ is called a *q-boundary*. Referring to the triangle $\langle v_i, v_j, v_k \rangle$ above, $\sigma = \langle v_i, v_j \rangle + \langle v_j, v_k \rangle + \langle v_i, v_k \rangle$ is both a 1-cycle and a 1-boundary since $\partial_1(\sigma) = 0$ and $\sigma = \partial_2(\langle v_i, v_j, v_k \rangle)$.

Two q -cycles a and a' are *homologous* if there exists a q -boundary b such that $a = a' + b$. Denote the groups of q -cycles and q -boundaries by $Z_q(S)$ and $B_q(S)$, respectively. All q -boundaries are q -cycles ($B_q(S) \subseteq Z_q(S)$). Define the *qth homology group* to be the quotient group $H_q(S) = Z_q(S)/B_q(S)$, for all q . Each element of $H_q(S)$ is a class $[a] = a + B_q(S)$ and a is a *representative q-cycle*. The *homology of S* is the collection of all the homology groups associated with S , i.e., $H_*(S) = \{H_q(S)\}_q$.

Let $(C_*(S), \partial)$ and $(C_*(S'), \partial')$ be chain complexes. A homomorphism $f = \{f_q : C_q(S) \rightarrow C_q(S')\}_q$ such that $f_q \partial_q = \partial'_q f_q$ for all q is a *chain map*. Note that the identity $id_{C_*(S)} = \{id_{C_q(S)} : C_q(S) \rightarrow C_q(S)\}_q$ is a chain map.

Let $f = \{f_q : C_q(S) \rightarrow C_q(S')\}_q$ and $g = \{g_q : C_q(S) \rightarrow C_q(S')\}_q$ be chain maps. A *chain homotopy from f to g* is a homomorphism $\phi = \{\phi_q : C_q(S) \rightarrow C_{q+1}(S')\}_q$ such that $\phi_{q-1} \partial_q + \partial'_{q+1} \phi_q = f_q + g_q$ for all q . A *chain contraction of $(C_*(S), \partial)$ to $(C_*(S'), \partial')$* is a triple $(f = \{f_q : C_q(S) \rightarrow C_q(S')\}_q, g = \{g_q : C_q(S') \rightarrow C_q(S)\}_q, \phi = \{\phi_q : C_q(S) \rightarrow C_{q+1}(S')\}_q)$ such that

- (i) f and g are chain maps;
- (ii) ϕ is a chain homotopy from $id_{C_*(S)}$ to $gf = \{g_q f_q : C_q(S) \rightarrow C_q(S')\}_q$;
- (iii) $fg = \{f_q g_q : C_q(S') \rightarrow C_q(S)\}_q = id_{C_*(S')}$.

Cochain groups are the linear duals of chain groups. Given a chain complex $(C_*(S), \partial)$, a *q-cochain* $c \in Hom(C_q(S), \mathbb{Z}/2)$. If we index the q -cells in a cellular complex from 1 to n_q , their corresponding duals generate $C^*(S)$. Thus a cochain $c \in C^*(S)$ is a \mathbb{Z}_2 -linear combination of the n_q elements in the dual basis, and as such can be thought of as a bit string of length n_q .

The set $C^q(S)$ of all q -cochains is a group, and the direct sum of all cochain groups associated with S is the graded group $C^*(S) = \{C^q(S)\}_q$. The *coboundary operator* $\delta = \{\delta^q : C^q(S) \rightarrow C^{q+1}(S)\}_q$ is defined on a q -cochain c by $\delta^q(c) = c \partial_{q+1}$. Note that $\delta \circ \delta = 0$. The associated *cochain complex* is the pair $(C^*(S), \delta)$.

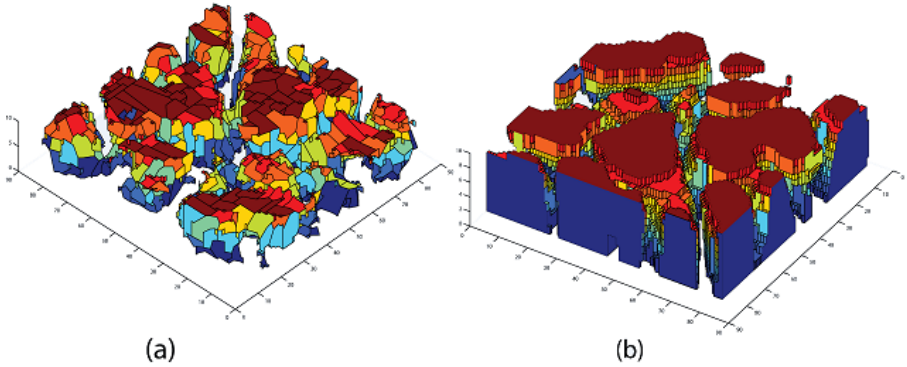


Fig. 7. (a) The cellular complex $\partial P(\mu I)$ for 10 edges as upper bound on $p \in \partial P(\mu I)$. (b) The cellular complex $\partial P(\mu I)$ preserving geometry.

A q -cochain c is a q -cocycle if $\delta^q(c) = 0$. A q -cochain b is a q -coboundary if there exists a $(q - 1)$ -cochain c such that $b = \delta^{q-1}(c)$. Two q -cocycles c and c' are *cohomologous* if there exists a q -coboundary b such that $c = c' + b$ (see Figure 8). We denote the subgroup of q -cocycles by $Z^q(S)$, and the subgroup of q -coboundaries by $B^q(S)$. The q^{th} *cohomology group* is defined to be the quotient $H^q(S) = Z^q(S)/B^q(S)$. Each element of $H^q(S)$ is a class $[c] = c + B^q(S)$. The element c is a *representative q -cocycle* of the cohomology class $[c]$. The *cohomology* of S is the graded $\mathbb{Z}/2$ -vector space $H^*(S) = \{H^q(S)\}_q$.

Since $P(I)$ (the output of Algorithm 2) is embedded in \mathbb{R}^3 , homology and cohomology of $P(I)$ are isomorphic and torsion free.

An *AT-model* [10,11] for a chain complex $(C_*(S), \partial)$, denoted by $((S, \partial), H, f, g, \phi)$, consists of a chain complex $(C_*(H), \partial' \equiv 0)$ together with a chain contraction (f, g, ϕ) of $(C_*(S), \partial)$ to $(C_*(H), \partial')$. The following properties hold:

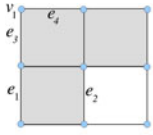
- If $\sigma \in H_q$, then $g_q(\sigma) \in C_q(S)$ is a representative cycle of a class of $H_q(C(S))$.
- The cochain $\partial_\sigma f : C_q(S) \rightarrow \mathbb{Z}/2$ defined by

$$\partial_\sigma f(\mu) := \begin{cases} 1, & \text{if } \sigma \text{ appears in the expression of } f(\mu), \\ 0, & \text{otherwise;} \end{cases}$$

is a representative cocycle of a class of $H^q(S)$.

- The map $H_q \rightarrow H_q(S)$ given by $\sigma \mapsto [g(\sigma)]$ linearly extends to an isomorphism $C_q(H) \cong H_q(S)$.
- The map $H_q \rightarrow H^q(S)$ given by $\sigma \mapsto [\partial_\sigma f]$ linearly extends to an isomorphism $C_q(H) \cong H^q(S)$.

Example 2. Consider the cellular complex $\partial P(\mu I)$ shown in Figure 7b obtained after applying Algorithm 2 to a μ MRI of a trabecular bone of size $85 \times 85 \times 10$



0-cochain	$\{v_1\}$
1-cochain	$\{e_1, e_4\}$
1-coboundary	$\delta\{v_1\} = \{e_3, e_4\}$
1-cocycle	$c = \{e_1, e_2\}$
1-cocycle	$d = \{e_1, e_2, e_3, e_4\}$
homologous cocycles	c and d ; since $d = c + \delta\{v_1\}$

Fig. 8. Example of cochain, cocycle and coboundary

voxels. Table 1 shows the results of the homology computation, i.e, the number of connected components, holes and cavities obtained after computing an AT-model for $\partial P(\mu I)$. Representative 1-cycles are shown in Figure 9.

Table 1. Results of the homology groups computation for the cellular complex $\partial P(\mu I)$ shown in Figure 7b. (see Figure 9).

Cellular Complex	H_0	H_1	H_2
$\partial P(\mu I)$	5	2	6

An AT-model for $(C_*(S), \partial)$ always exists and can be computed in $\mathcal{O}(m^3)$, where m is the number of elements of S (see [10,11]).

Given an AT-model $((P(I), \partial), H, f, g, \phi)$ for $P(I)$ (the output of Algorithm 2), we have that

$$C_*(H) \cong H_*(P(I)) \cong H^*(P(I)) \cong Hom(H_*(P(I)), \mathbb{Z}/2) \cong C^*(H).$$

Let $\alpha \in H_n$. Consider the dual elementary n -cocycle in $C^n(H)$,

$$\alpha^* : C_n(H) \rightarrow \mathbb{Z}/2 \quad \text{such that for } \mu \in H_n, \quad \alpha^*(\mu) := \begin{cases} 1 & \text{if } \mu = \alpha, \\ 0 & \text{otherwise.} \end{cases}$$

Proposition 1. Given an ordering $\{v_1 < \dots < v_n\}$ of the vertices of $P(I)$, each polygon $p \in P(I)$ can be expressed as an ordered list of vertices $\{v_{i_1} < \dots < v_{i_k}\} \subseteq \{v_1 < \dots < v_n\}$ with edges

$$e_j := \begin{cases} \langle v_{i_j}, v_{i_{j+1}} \rangle, & \text{if } j < k, \\ \langle v_{i_k}, v_{i_1} \rangle, & \text{if } j = k. \end{cases}$$

The following two theorems formulate a diagonal approximation ∇' on a polygon and the cup product on $H^*(P(I))$ in terms of ∇' . All non-trivial cup products in $H^*(P(I))$ are products of 1-cocycles for dimensional reasons.

Theorem 1. Consider a polygon $p = \langle v_1, \dots, v_n \rangle$ with edges $e_i = \langle v_i, v_{i+1} \rangle$, $i < n$, and $e_n = \langle v_n, v_1 \rangle$. Then a diagonal approximation on p is given by

$$\nabla'(p) := \sum_{1 < i < n, v_i < v_{i+1}} (e_1 + \dots + e_{i-1}) \otimes e_i + \sum_{1 < i < n, v_i > v_{i+1}} (e_{i+1} + \dots + e_n) \otimes e_i.$$

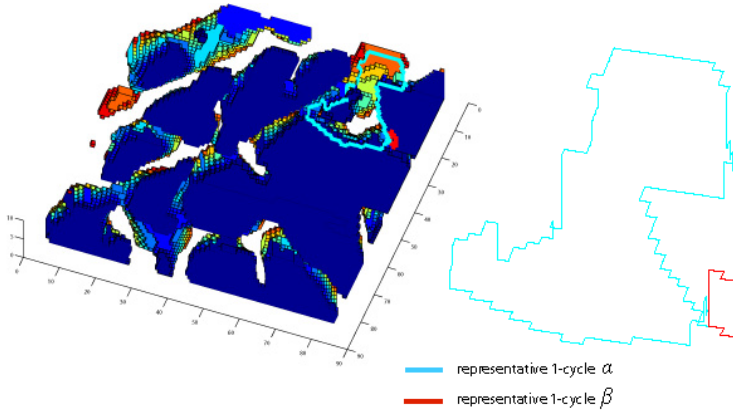


Fig. 9. Representative 1-cycles

Theorem 2. Consider a digital image I , the cellular complex $P(I)$, an ordering of the vertices of $P(I)$, and an AT-model $((P(I), \partial), H, f, g, \phi)$ for $P(I)$. Then for $\alpha, \beta \in H_1$ and $\gamma \in H_2$ the cup product $\alpha^* \smile' \beta^*$ is given by

$$(\alpha^* \smile' \beta^*)(\gamma) = m(\partial_\alpha f \otimes \partial_\beta f) \nabla' g(\gamma), \tag{1}$$

where m denotes multiplication in $\mathbb{Z}/2$. The cup product is bilinear, commutative, associative, and independent of the ordering of the vertices .

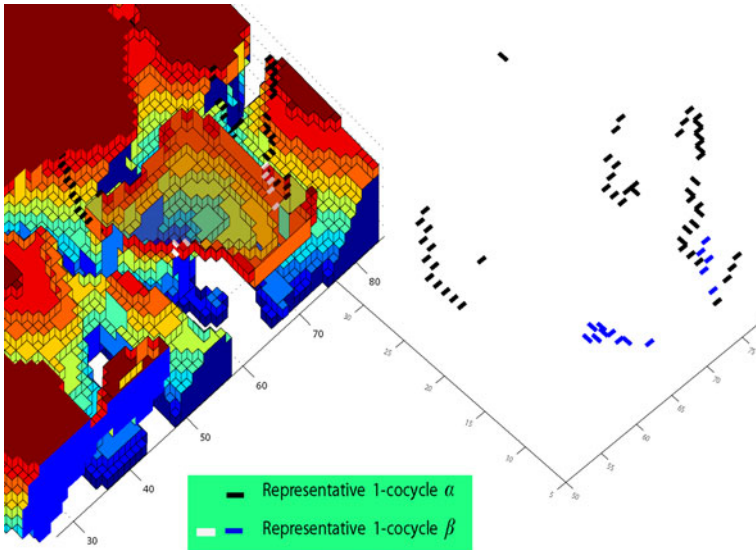


Fig. 10. Representative 1-cocycles

Table 2. Results of the computation of the cup product for the cellular complex shown in Figure 7b

	(α, α)	(α, β)	(β, β)
γ_1	0	0	0
γ_2	0	0	0
γ_3	0	0	0
γ_4	0	0	0
γ_5	0	0	0
γ_6	0	1	0

Table 3. Time (in seconds) and the results of the computation of the cup product on the cellular complexes shown in Figure 11

Cell complex	Number of 2-cells	Time to compute the cup product				
A (see Figure 11)	1920	18.19 sec.				
B (see Figure 11)	57	3.8 sec.				

	(α_1, α_2)	(α_1, α_3)	(α_1, α_4)	(α_2, α_3)	(α_2, α_4)	(α_3, α_4)
β	0	0	1	1	0	0

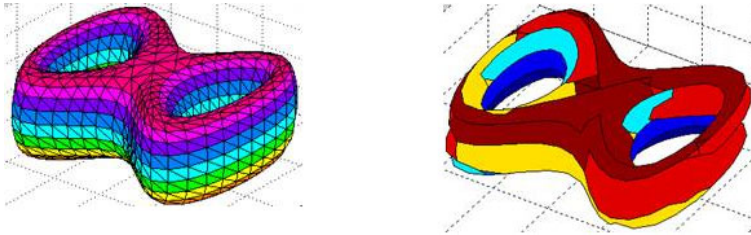


Fig. 11. Left: cell complex A. Right: cell complex B

Proof. (Sketch) To verify formula (II), we subdivide the polygons of $P(I)$, obtain a very special triangulation K of $P(I)$, and apply a chain contraction (f_T, g_T, ϕ_T) of $C_*(K)$ to $C_*(P(I))$ (we only need to subdivide the polygons of $P(I)$ since the cup product is non-trivial only on 1-cocycles). We then appeal to the standard formula on a triangle with vertices $v_i < v_j < v_k$ (see [17]):

$$\nabla(\langle v_i, v_j, v_k \rangle) = \langle v_i, v_j \rangle \otimes \langle v_j, v_k \rangle.$$

Finally, for a polygon $p \in P(I)$ we have $\nabla'(p) = (f_T \otimes f_T)\nabla g_T(p)$. □

Example 3. Starting from the results obtained in Example 2 and applying the formula given in Theorem 2, representative 1-cocycles are shown in Figure 10. Table 2 shows the cup product.

The following table illustrates the dramatic improvement in computational efficiency realized by removing faces and non-critical vertices:

4 Conclusions and Plans for Future Work

Given a 3D digital image I , we have formulated the cup product on the cohomology of the cellular complex $P(I)$ obtained by simplifying the cubical complex $Q(I)$. The algorithm proposed here is valid for any encoding of a 3D digital object given as a set of polyhedra. Nevertheless, our ultimate goal is to compute the cup product on any cellular complex without making use of triangulations or other kind of subdivision. To this end, we shall apply some standard geometric constructions such as forming quotients, taking Cartesian products, and merging cells.

References

1. Argawal, P.K., Suri, S.: Surface approximation and geometric partitions. *SIAM Journal on Computing* 27(4), 1016–1035 (1998)
2. Brönnimann, H., Goodrich, M.T.: Almost optimal set covers in finite VC-dimension. *Discrete and Computational Geometry* 14, 263–279 (1995)
3. Das, G., Goodrich, M.T.: On the complexity of optimization problem for 3D convex polyhedra and decision trees. *Computational Geometry: Theory and Applications* 8, 123–137 (1997)
4. Hatcher, A.: *Algebraic Topology*. Cambridge University Press, Cambridge (2002)
5. Gonzalez-Diaz, R., Ion, A., Iglesias-Ham, M., Kropatsch, W.: Irregular graph pyramids and representative cocycles of cohomology generators. In: Torsello, A., Escolano, F., Brun, L. (eds.) *GbRPR 2009*. LNCS, vol. 5534, pp. 263–272. Springer, Heidelberg (2009)
6. Gonzalez-Diaz, R., Ion A., Iglesias-Ham M., Kropatsch W.: Invariant representative cocycles of cohomology generators using irregular graph pyramids. *Computer Vision and Image Understanding* (in press)
7. Gonzalez-Diaz, R., Jimenez, M.J., Medrano, B., Molina-Abril, H., Real, P.: Integral operators for computing homology generators at any dimension. In: Ruiz-Shulcloper, J., Kropatsch, W.G. (eds.) *CIARP 2008*. LNCS, vol. 5197, pp. 356–363. Springer, Heidelberg (2008)
8. Gonzalez-Diaz, R., Jimenez, M.J., Medrano, B.: Cohomology ring of 3D cubical complexes. In: *IWCIA 2009*. LNCS, vol. 5852, pp. 139–150. Springer, Heidelberg (2009)
9. Gonzalez-Diaz, R., Jimenez, M.J., Medrano, B.: Cohomology ring of 3D photographs. *Int. Journal of of Imaging Systems and Technology* (in press)
10. Gonzalez-Diaz, R., Real, P.: Towards digital cohomology. In: Nyström, I., Sanniti di Baja, G., Svensson, S. (eds.) *DGCI 2003*. LNCS, vol. 2886, pp. 92–101. Springer, Heidelberg (2003)

11. Gonzalez-Diaz, R., Real, P.: On the cohomology of 3D digital images. *Discrete Applied Math.* 147, 245–263 (2005)
12. Kaczynski, T., Mischaikow, K., Mrozek, M.: *Computational homology*. Applied Mathematical Sciences 157 (2004)
13. Kovalevsky, V.A., Schulz, H.: Convex hulls in a 3D space. In: Klette, R., Žunić, J. (eds.) *IWCIA 2004*. LNCS, vol. 3322, pp. 176–196. Springer, Heidelberg (2004)
14. Kravatz, D.: Diagonal approximations on an n-gon and the cohomology ring of closed compact orientable surfaces. Senior Thesis. Millersville University Department of Mathematics (2008)
15. Lamar-Leon, J., Garcia-Reyes, E., Gonzalez-Diaz, R.: Human gait recognition using topological information. In: *Proc. of CTIC (2010)*; *Electronic Journal Image-A*, <http://munkres.us.es/> 1(3) (2010). Selected paper invited to submit an extended version to a Special issue devoted to CTIC2010 in *Pattern Recognition Letter*
16. Latecki, L.J.: 3D well-composed pictures. *Graphical Models and Image Processing* 59(3), 164–172 (1997)
17. Munkres, J.R.: *Elements of Algebraic Topology*. Addison-Wesley Co., Reading (1984)
18. Peltier, S., Ion, A., Kropatsch, W.G., Damiand, G., Haxhimusa, Y.: Directly computing the generators of image homology using graph pyramids. *Image and Vision Computing* 27(7), 846–853 (2009)
19. Schulz, H.: Polyhedral approximation and practical convex hull algorithm for certain classes of voxel sets. *Discrete Appl. Math.* 157(16), 3485–3493 (2009)
20. Serre, J.P.: Homologie singuliere des espaces fibrés, applications. *Ann. Math.* 54, 429–501 (1951)

A Jordan Curve Theorem in the Digital Plane

Josef Slapal

Brno University of Technology, Department of Mathematics,
616 69 Brno, Czech Republic
slapal@fme.vutbr.cz

Abstract. We study a certain Alexandroff topology on \mathbb{Z}^2 and some of its quotient topologies including the Khalimsky one. By proving an analogue of the Jordan curve theorem for this topology we show that it provides a large variety of digital Jordan curves. Some consequences of this result are discussed, too.

1 Introduction

Geometric and topological properties of (two-dimensional) digital images are crucial in creating new, efficient algorithms to solve various problems of computer image processing. To study these properties, we need the digital plane \mathbb{Z}^2 to be equipped with a convenient structure. Here, the convenience means that such a structure satisfies some analogues of basic geometric and topological properties of the Euclidean topology on \mathbb{R}^2 . Most importantly, it is usually required that an analogue of the Jordan curve theorem be valid. (Recall that the classical Jordan curve theorem states that any simple closed curve in the Euclidean plane separates this plane into exactly two components). In the classical approach to this problem (see e.g. [11] and [12]), graph theoretic tools are used for structuring \mathbb{Z}^2 , namely the well-known binary relations of 4-adjacency and 8-adjacency. Unfortunately, neither 4-adjacency nor 8-adjacency itself allows an analogue of the Jordan curve theorem - cf. [8]. To overcome this, a combination of the two binary relations has to be used. Despite this inconvenience, the graph-theoretic approach is used to solve many problems of digital image processing and to create useful graphic software. In [5], a new, purely topological approach to the problem was proposed which utilizes a convenient topology on \mathbb{Z}^2 , called Khalimsky topology (cf. [4]), for structuring the digital plane. At present, this topology is one of the most important concepts of the theory called digital topology. It has been studied and used by many authors, see e.g. [2] and [6]-[9]. The possibility of employing convenient topological structures on \mathbb{Z}^2 different from the Khalimsky topology is discussed in [13]-[17].

In [17], a new topology on \mathbb{Z}^2 has been introduced and studied that was obtained by a slight modification of the topology introduced in [15] and then studied also in [16]. It was shown in [17] that the new topology provides a certain convenient Jordan curves behaving more advantageously than the Jordan curves in the Khalimsky space. And, by results proved in [16], the quotient topologies of this topology include the Khalimsky topology as well as two other convenient

topologies on \mathbb{Z}^2 . In the present note we continue the study from [15]. We will show that the variety of Jordan curves in the topology under consideration is much larger than that one determined in [15].

2 Preliminaries

For the topological terminology used we refer to [1] and [3]. Throughout the note, all topologies dealt with are thought of as being (given by) Kuratowski closure operators. Recall that a topology p on a set X is said to be a T_0 -topology if, for arbitrary points $x, y \in X$, from $x \in p\{y\}$ and $y \in p\{x\}$ it follows that $x = y$, and it is called a $T_{\frac{1}{2}}$ -topology if each singleton subset of X is closed or open (so that $T_{\frac{1}{2}}$ implies T_0). Recall also that p is said to be an Alexandroff topology if $pA = \bigcup_{x \in A} p\{x\}$ whenever $A \subseteq X$. So, if p is an Alexandroff topology on a set X , then it is given by determining the closures of all points of X and there is an Alexandroff topology \bar{p} on X given by $x \in \bar{p}\{y\} \Leftrightarrow y \in p\{x\}$ whenever $x, y \in X$. The topology \bar{p} is said to be dual to p . Clearly, $\bar{\bar{p}} = p$ and a subset $A \subseteq X$ is closed (open) in (X, p) if and only if it is open (closed) in (X, \bar{p}) .

A map $f : (X, p) \rightarrow (Y, q)$ between topological spaces (X, p) and (Y, q) is said to be continuous if $f(pA) \subseteq q(f(A))$ whenever $A \subseteq X$. Given a topological space (X, p) and a surjection $e : X \rightarrow Y$, a topology q on Y is called the quotient topology of p generated by e if q is the finest topology on Y for which $e : (X, p) \rightarrow (Y, q)$ is continuous. Here, given topologies q and r on X , q is said to be finer than r (and r coarser than q) if $qA \subseteq rA$ for every $A \subseteq X$.

By a graph on a set V we always mean an undirected simple graph without loops whose vertex set is V . Recall that a path in a graph is a finite (nonempty) sequence x_0, x_1, \dots, x_n of pairwise different vertices such that x_{i-1} and x_i are adjacent (i.e., joined by an edge) whenever $i \in \{1, 2, \dots, n\}$. By a cycle in a graph we understand any finite set of at least three vertices which can be ordered into a path whose first and last members are adjacent.

The connectedness graph of a topology p on X is the graph on X in which a pair of vertices x, y is adjacent if and only if $x \neq y$ and $\{x, y\}$ is a connected subset of (X, p) . Let p be an Alexandroff topology on a set X . Then a subset $A \subseteq X$ is connected in (X, p) if and only if each pair of points of A may be joined by a path in the connectedness graph of (X, p) contained in A . Clearly, p is given by its connectedness graph provided that every edge of the graph is adjacent to a point which is known to be closed or to a point which is known to be open (in which case p is T_0). Indeed, the closure of a closed point consists of just this point, the closure of an open point consists of this point and all points adjacent to it and the closure of a mixed point (i.e., a point that is neither closed nor open) consists of this point and all closed points adjacent to it. In the sequel, only the connectedness graphs of some connected Alexandroff topologies on \mathbb{Z}^2 will be considered in which the closed points will be ringed and the mixed ones boxed (so that the points neither ringed nor boxed will be open - note that no points of \mathbb{Z}^2 may be both closed and open). Obviously, there are 2^{\aleph_0} Alexandroff T_0 -pretopologies on \mathbb{Z}^2 having the same given connectedness graph.

By a (*discrete*) *closed curve* in a topological space (X, p) we mean a cycle in the connectedness graph of p . Thus, every cycle is a nonempty, finite and connected set. In accordance with [16], a closed curve $C \subseteq X$ in (X, p) is said to be *simple* if, for each point $x \in C$, there are exactly two points of C adjacent to x in the connectedness graph of p . A simple closed curve C in (X, p) is said to be a (*discrete*) *Jordan curve* if it separates (X, p) into precisely two components (i.e., if the subspace $X - C$ of (X, p) consists of precisely two components).

Since we will work with quotient topologies of a certain Alexandroff topology on \mathbb{Z}^2 , we will start with presenting some general facts concerning the behavior of quotient topologies of Alexandroff topologies.

Lemma 1. *Let (X, p) be an Alexandroff space, let Y be a set and let $e : X \rightarrow Y$ be a surjection. Then the following condition is necessary and sufficient for a topology q on Y to be the quotient topology of p generated by e :*

q is an Alexandroff topology on Y with the property that, for every pair of points $x, y \in Y$, $x \in q\{y\}$ if and only if there are $a \in e^{-1}(x)$ and $b \in e^{-1}(y)$ such that $a \in p\{b\}$.

Proof. Let q be the quotient topology of p generated by e and let $x, y \in Y$. If there are $a \in e^{-1}(x)$ and $b \in e^{-1}(y)$ such that $a \in p\{b\}$, then $x \in q\{y\}$ because $e : (X, p) \rightarrow (Y, q)$ is continuous. The converse implication follows from the fact that q is the smallest topology on Y such that $e : (X, p) \rightarrow (Y, q)$ is continuous. The same fact implies that q is Alexandroff.

Conversely, let the condition of the statement be satisfied. If $a, b \in X$ are points such that $a \in p\{b\}$, then, putting $x = e(a)$ and $y = e(b)$, we get $a \in e^{-1}(x)$ and $b \in e^{-1}(y)$. Therefore, $e(a) = x \in q\{y\} = q\{e(b)\}$. We have shown that $e : (X, p) \rightarrow (Y, q)$ is continuous. Let q' be an arbitrary topology on Y such that $e : (X, p) \rightarrow (Y, q')$ is continuous. Let $B \subseteq Y$ be a subset and $x \in qB$ be a point. Then there is a point $y \in B$ with $x \in q\{y\}$. Consequently, there are points $a \in e^{-1}(x)$ and $b \in e^{-1}(y)$ such that $a \in p\{b\}$. We get $a \in p\{b\} \subseteq p\{e^{-1}(y)\} \subseteq p\{e^{-1}(B)\}$, hence $x = e(a) \in q'(e(e^{-1}(B))) = q'B$. We have shown that $qB \subseteq q'B$. Thus, $q \leq q'$ and the statement is proved.

Proposition 1. *Let (X, p) be an Alexandroff space, $e : X \rightarrow Y$ be a surjection and let q be the quotient topology of p on Y generated by e . Let e have the property that $e^{-1}(\{y\})$ is connected in (X, p) for every point $y \in Y$ and let $B \subseteq Y$ be a subset. Then B is connected in (Y, q) if and only if $e^{-1}(B)$ is connected in (X, p) .*

Proof. If $e^{-1}(B)$ is connected, then so is B because $B = e(e^{-1}(B))$. Conversely, let B be connected. Then every pair of points of B may be joined by a path in the connectedness graph of q contained in B . As $e^{-1}(y)$ is connected for every point $y \in B$, Lemma 1 implies that every pair of points of $e^{-1}(B) = \bigcup_{y \in B} e^{-1}(y)$ may be joined by a path in the connectedness graph of p contained in $e^{-1}(B)$. Thus, $e^{-1}(B)$ is connected in (X, p) .

Let us note that, in general, for a topological space (X, p) and a quotient topology of p , the statement of the previous Lemma need not be true. It is true if, for example, B is closed or open in (Y, q) .

Corollary 1. *Let (X, p) be an Alexandroff space and let q be the quotient topology of p on a set Y generated by a surjection $e : X \rightarrow Y$. Let e have the property that $e^{-1}(\{y\})$ is connected for every point $y \in Y$ and let $C \subseteq X$ be a simple closed curve in (X, p) . Then C is a Jordan curve in (X, p) if the following two conditions are satisfied:*

- (1) $e(C)$ is a Jordan curve in (Y, q) , i.e., separates (Y, q) into exactly two components D_1 and D_2 .
- (2) The subspace $e^{-1}(e(C)) - C$ of (X, p) has at most two components and there are sets E_1, E_2 with $\{E_1, E_2\} = \{C_1, C_2\}$ if $e^{-1}(e(C)) - C$ has two components C_1, C_2 and $\{E_1, E_2\} = \{\emptyset, C_0\}$ if $e^{-1}(e(C)) - C$ has only one component C_0 (in which case $C_0 = e^{-1}(e(C)) - C$) such that, for every $i \in \{1, 2\}$, $E_i \neq \emptyset$ implies that there is a point in C_i adjacent to a point of $e^{-1}(D_i)$ but no point in C_i is adjacent to a point of $e^{-1}(D_{3-i})$ (in the connectedness graph of p).

Proof. Let the conditions (1) and (2) of the statement be fulfilled and, for every $i \in \{1, 2\}$, put $C'_i = E_i \cup e^{-1}(D_i)$. Clearly, $C'_1 \cup C'_2 = X - C$. By Proposition 1, $e^{-1}(D_i)$ is connected, hence C'_i is connected for both $i = 1, 2$. Since $C'_1 \cup C'_2$ is clearly not connected, C'_1 and C'_2 are the components of $X - C$.

Let $z = (x, y) \in \mathbb{Z}^2$ be a point. We put

$$\begin{aligned} H_2(z) &= \{(x + k, y); k \in \{-1, 1\}\}, \\ V_2(z) &= \{(x, y + l); l \in \{-1, 1\}\}, \\ D_5(z) &= H_2(z) \cup \{(x + k, y - 1); k \in \{-1, 0, 1\}\}, \\ U_5(z) &= H_2(z) \cup \{(x + k, y + 1); k \in \{-1, 0, 1\}\}, \\ L_5(z) &= V_2(z) \cup \{(x - 1, y + l); l \in \{-1, 0, 1\}\}, \\ R_5(z) &= V_2(z) \cup \{(x + 1, y + l); l \in \{-1, 0, 1\}\}. \end{aligned}$$

Next, we put

$$\begin{aligned} A_4(z) &= H_2(z) \cup V_2(z), \\ A_8(z) &= L_5(z) \cup R_5(z) (= D_5(z) \cup U_5(z)), \text{ and} \\ A'_4(z) &= A_8(z) - A_4(z). \end{aligned}$$

Thus, the number of points of each of the nine sets introduced above equals the index of the symbol denoting this set. In the literature, the points of $A_4(z)$ and $A_8(z)$ are said to be *4-adjacent* and *8-adjacent* to z , respectively. It is natural to call the points of $H_2(z)$, $V_2(z)$, $D_5(z)$, $U_5(z)$, $L_5(z)$, $R_5(z)$ and $A'_4(z)$ *horizontally 2-adjacent*, *vertically 2-adjacent*, *down 5-adjacent*, *up 5-adjacent*, *left 5-adjacent*, *right 5-adjacent* and *diagonally 4-adjacent* to z , respectively. Clearly, each of these adjacencies implies 8-adjacency.

The union of each of the above nine sets $H_2(z)$, $V_2(z)$... with the singleton $\{z\}$ is denoted by the corresponding bared symbols, i.e., by $\bar{H}_2(z)$, $\bar{V}_2(z)$

Recall [5] that the Khalimsky topology on \mathbb{Z}^2 is the Alexandroff topology t given as follows:

For any $z = (x, y) \in \mathbb{Z}^2$,

$$t\{z\} = \begin{cases} \bar{A}_8(z) & \text{if } x, y \text{ are even,} \\ \bar{H}_2(z) & \text{if } x \text{ is even and } y \text{ is odd,} \\ \bar{V}_2(z) & \text{if } x \text{ is odd and } y \text{ is even,} \\ \{z\} & \text{otherwise.} \end{cases}$$

The Khalimsky topology is connected and T_0 ; a portion of its connectedness graph is shown in Figure 1.

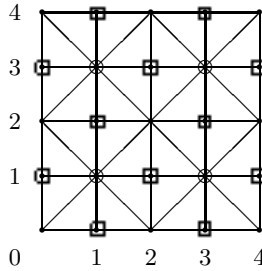


Fig. 1. A portion of the connectedness graph of the Khalimsky topology

In the literature, the topology \bar{t} dual to t is also called Khalimsky.

Clearly, (\mathbb{Z}^2, t) is a connected topological space - it is called the *Khalimsky plane*. In [5], Khalimsky, Koppermann and Meyer proved the following digital Jordan curve theorem for (\mathbb{Z}^2, t) :

Theorem 1. *In the Khalimsky plane, any simple closed curve having at least four points is a Jordan curve.*

We will need the following

Corollary 2. *Let C be a closed curve in the Khalimsky plane such that every point $z \in \mathbb{Z}^2$ with $z = (2k, 2l + 1)$ or $z = (2k + 1, 2l)$ for some $k, l \in \mathbb{Z}$ satisfies the following two conditions:*

- (1) $z \in C$ implies that $H_2(z) \subseteq C$ or $V_2(z) \subseteq C$.
- (2) If $A \subseteq C \cap A_4(z)$ is a three-element set, then $z \notin C$ and there are two points in A that are the only points of C adjacent to the other point in A (in the adjacency graph of the Khalimsky topology).

Then C is a Jordan curve in the Khalimsky plane.

Proof. The statement follows from Theorem 1 because it may easily be shown that, in the Khalimsky plane, the closed curves satisfying conditions (1) and (2) coincide with the simple closed curves having at least four points.

Note that simple closed curves in the Khalimsky plane can not turn, at any of its points, at the acute angle $\frac{\pi}{4}$ and, in mixed points, they can not turn at all.

3 Topology w

We denote by w the Alexandroff topology on \mathbb{Z}^2 given as follows:

For any point $z = (x, y) \in \mathbb{Z}^2$,

$$w\{z\} = \begin{cases} \bar{A}_8(z) & \text{if } x = 4k, y = 4l, k, l \in \mathbb{Z}, \\ \bar{A}'_4(z) & \text{if } x = 2 + 4k, y = 2 + 4l, k, l \in \mathbb{Z}, \\ \bar{D}_5(z) & \text{if } x = 2 + 4k, y = 1 + 4l, k, l \in \mathbb{Z}, \\ \bar{U}_5(z) & \text{if } x = 2 + 4k, y = 3 + 4l, k, l \in \mathbb{Z}, \\ \bar{L}_5(z) & \text{if } x = 1 + 4k, y = 2 + 4l, k, l \in \mathbb{Z}, \\ \bar{R}_5(z) & \text{if } x = 3 + 4k, y = 2 + 4l, k, l \in \mathbb{Z}, \\ \bar{H}_2(z) & \text{if } x = 2 + 4k, y = 4l, k, l \in \mathbb{Z}, \\ \bar{V}_2(z) & \text{if } x = 4k, y = 2 + 4l, k, l \in \mathbb{Z}, \\ \{z\} & \text{otherwise.} \end{cases}$$

Clearly, w is connected and T_0 . A portion of the connectedness graph of w is shown in Figure 2.

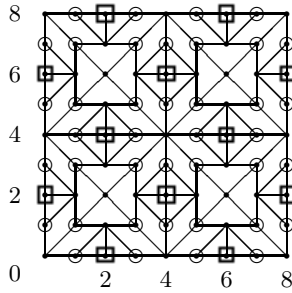


Fig. 2. A portion of the connectedness graph of w

The following result immediately follows from [15], Theorem 11:

Theorem 2. *Every cycle in the graph a portion of which is demonstrated in Figure 3 is a Jordan curve in (\mathbb{Z}^2, w) .*

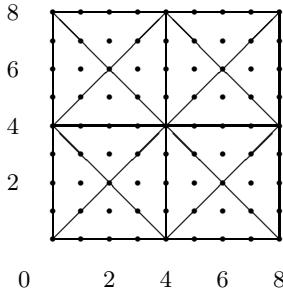


Fig. 3. A portion of a subgraph of the connectedness graph of w

We will need the following immediate consequence of [16], Theorem 10:

Theorem 3. *The Khalimsky topology is the quotient topology of w generated by the surjection $f : \mathbb{Z}^2 \rightarrow \mathbb{Z}^2$ given as follows:*

$$f(x, y) = \begin{cases} (2k, 2l) & \text{if } (x, y) = (4k, 4l), \quad k, l \in \mathbb{Z}, \\ (2k, 2l + 1) & \text{if } (x, y) \in \bar{A}_4(4k, 4l + 2), \quad k, l \in \mathbb{Z}, \\ (2k + 1, 2l) & \text{if } (x, y) \in \bar{A}'_4(4k + 2, 4l), \quad k, l \in \mathbb{Z}, \\ (2k + 1, 2l + 1) & \text{if } (x, y) \in \bar{A}'_4(4k + 2, 4l + 2), \quad k, l \in \mathbb{Z}. \end{cases}$$

The surjection f is demonstrated in Figure 4 where the corresponding decomposition of the topological space $(\mathbb{Z}^2 \times \mathbb{Z}^2, w)$ is marked by the dashed lines. All points of a class of the decomposition are mapped by f to the center point of the class given by the bold coordinates.

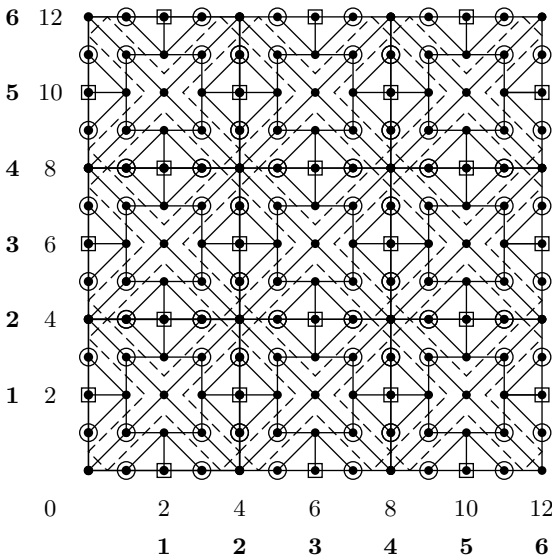


Fig. 4. Decomposition of $(\mathbb{Z} \times \mathbb{Z}, w)$ given by the surjection f

Consider the following three conditions for a cycle C in the topological space (\mathbb{Z}^2, w) :

- (1) $\bar{H}_2(z) \not\subseteq C$ whenever $z = (4k + 2, 2l + 1)$ for some $k, l \in \mathbb{Z}$ and $\bar{V}_2(z) \not\subseteq C$ whenever $z = (2k + 1, 4l + 2)$ for some $k, l \in \mathbb{Z}$.
- (2) If $z \in \mathbb{Z}^2$ is a point with $z = (4k, 4l + 2)$ for some $k, l \in \mathbb{Z}$, then
 - a) $C \cap A_4(z) \neq \emptyset$ implies that either $\{(4k, 4l), (4k, 4l + 4)\} \subseteq C$ or $C \cap V_2(4k - 1, 4l + 2) \neq \emptyset \neq C \cap V_2(4k + 1, 4l + 2)$ and
 - b) if $\{(4k, 4l), (4k, 4l + 4)\} \subseteq C$ and $C \cap (V_2(4k - 1, 4l + 2) \cup V_2(4k + 1, 4l + 2)) = A$ where A is a singleton, then $V_2(4k - 1, 4l + 2) \cup \{(4k - 2, 4l + 2)\} \subseteq C$ if $A \subseteq V_2(4k - 1, 4l + 2)$ and $V_2(4k + 1, 4l + 2) \cup \{(4k + 2, 4l + 2)\} \subseteq C$ if $A \subseteq V_2(4k + 1, 4l + 2)$.

- (3) If $z \in \mathbb{Z}^2$ is a point with $z = (4k + 2, 4l)$ for some $k, l \in \mathbb{Z}$, then
- a) $C \cap A_4(z) \neq \emptyset$ implies that either $\{(4k, 4l), (4k + 4, 4l)\} \subseteq C$ or $C \cap H_2(4k + 2, 4l - 1) \neq \emptyset \neq C \cap H_2(4k + 2, 4l + 1)$ and
 - b) if $\{(4k, 4l), (4k + 4, 4l)\} \subseteq C$ and $C \cap (H_2(4k + 2, 4l - 1) \cup H_2(4k + 2, 4l + 1)) = A$ where A is a singleton, then $H_2(4k + 2, 4l - 1) \cup \{(4k + 2, 4l - 2)\} \subseteq C$ if $A \subseteq H_2(4k + 2, 4l - 1)$ and $H_2(4k + 2, 4l + 1) \cup \{(4k + 2, 4l + 2)\} \subseteq C$ if $A \subseteq H_2(4k + 2, 4l + 1)$.

As the main result of this note, we get the following digital analogue of the Jordan curve theorem which generalizes Theorem 2:

Theorem 4. *In the topological space (\mathbb{Z}^2, w) , every simple closed curve C with at least eight points satisfying conditions (1)-(3) is a Jordan curve.*

Proof. It may easily be seen that every simple closed curve C in (\mathbb{Z}^2, w) with at least eight points satisfying conditions (1)-(3) has the property that its image $D = f(C)$, where f is the surjection from Theorem 3, is a cycle in the Khalimsky plane fulfilling the assumptions of Corollary 2. Therefore, the condition (1) of Corollary 1 is satisfied. It may be shown that condition (2) of Corollary 1 is satisfied too, which proves the statement.

Remark 1. Clearly, there are Jordan curves in (\mathbb{Z}^2, w) that have not been determined in Theorem 4. These are, for example, the "square" Jordan curves having four points and also those having six points (one may easily identify these Jordan curves among the closed ones in the connectedness graph of w).

Example 1. Every cycle in each of the six graphs portions of which are demonstrated in Figure 5 is a Jordan curve in (\mathbb{Z}^2, w) .

We have determined Jordan curves in the topological space (\mathbb{Z}^2, w) by using Jordan curves in a quotient space of (\mathbb{Z}^2, w) , namely the Khalimsky plane. Now, conversely, the Jordan curves in (\mathbb{Z}^2, w) may be used to determine Jordan curves in the topologies on \mathbb{Z}^2 that are quotient topologies of w . More precisely, we may apply the following statement:

Corollary 3. *Let (\mathbb{Z}^2, p) be an Alexandroff topological space and let q be the quotient topology of p on \mathbb{Z}^2 generated by a surjection $e : \mathbb{Z}^2 \rightarrow \mathbb{Z}^2$. Let e have the property that $e^{-1}(\{y\})$ is connected for every point $y \in \mathbb{Z}^2$ and let $D \subseteq \mathbb{Z}^2$ be a simple closed curve in (\mathbb{Z}^2, q) . Then D is a Jordan curve in (\mathbb{Z}^2, q) if the following two conditions are satisfied:*

- (1) *There is a Jordan curve C in (\mathbb{Z}^2, p) such that $e(C) = D$.*
- (2) *$C_i - e^{-1}(D)$ is nonempty and connected in (\mathbb{Z}^2, p) for $i = 1, 2$ where C_1 and C_2 are the two components of $\mathbb{Z}^2 - C$.*

Proof. Let the conditions of the statement be fulfilled and put $C'_1 = C_1 - e^{-1}(D)$ and $C'_2 = C_2 - e^{-1}(D)$. We clearly have $e(C_1) \cap e(C_2) = \emptyset$ (because, otherwise, there is a point $y \in e(C_1) \cap e(C_2)$, which means that $e^{-1}(\{z\}) \cap C_1 \neq \emptyset \neq e^{-1}(\{z\}) \cap C_2$ - this is a contradiction as $e^{-1}(\{z\})$ is connected).

Therefore, $e(C'_1) \cap e(C'_2) = \emptyset$. This yields $C'_i = e^{-1}(e(C'_i))$ for $i = 1, 2$, hence $e(C'_i)$ is connected for $i = 1, 2$ by Proposition 1. Suppose that $\mathbb{Z}^2 - D$ is connected. Then $e^{-1}(\mathbb{Z}^2 - D) = C'_1 \cup C'_2$ is connected by Proposition 1. This is a contradiction because $\emptyset \neq C'_i \subseteq C_i$ for $i = 1, 2$, C_1 and C_2 are disjoint and $C_1 \cup C_2$ is not connected. Therefore, $\mathbb{Z}^2 - D = e(C'_1) \cup e(C'_2)$ is not connected and, consequently, $e(C'_1)$ and $e(C'_2)$ are components of $\mathbb{Z}^2 - D$.

One of the quotient topologies of w is the Marcus-Wyse topology (cf. [10]), i.e., the connected Alexandroff $T_{\frac{1}{2}}$ -topology s on \mathbb{Z}^2 given as follows:

For any $z = (x, y) \in \mathbb{Z}^2$,

$$s\{z\} = \begin{cases} \bar{A}_4(z) & \text{if } x + y \text{ is odd,} \\ \{z\} & \text{otherwise.} \end{cases}$$

For the definition of the map $g : (\mathbb{Z}^2, w) \rightarrow (\mathbb{Z}^2, s)$ with respect to which s is the quotient topology of w see [16]. The Marcus-Wyse topology is quite simple - its connectedness graph coincides with the 4-adjacency graph. We will therefore discuss another quotient topology of w on \mathbb{Z}^2 .

Let v be the Alexandroff topology on \mathbb{Z}^2 given as follows:

For any $z = (x, y) \in \mathbb{Z}^2$,

$$v\{z\} = \begin{cases} \bar{H}_2(z) & \text{if } x \text{ is odd and } y \text{ is even,} \\ \bar{V}_2(z) & \text{if } x \text{ is even and } y \text{ is odd,} \\ \bar{A}'_4(z) & \text{if } x, y \text{ are odd,} \\ \{z\} & \text{if } x, y \text{ are even.} \end{cases}$$

Evidently, v is connected and $T_{\frac{1}{2}}$. A portion of its connectedness graph is shown in Figure 6.

Theorem 12 in [16] implies:

Theorem 5. v is the quotient topology of w generated by the surjection $h : \mathbb{Z}^2 \rightarrow \mathbb{Z}^2$ given as follows:

$$h(x, y) = \begin{cases} (2k, 2l) & \text{if } (x, y) \in \bar{A}_8(4k, 4l), \quad k, l \in \mathbb{Z}, \\ (2k, 2l + 1) & \text{if } (x, y) \in \bar{H}_2(4k, 4l + 2), \quad k, l \in \mathbb{Z}, \\ (2k + 1, 2l) & \text{if } (x, y) \in \bar{V}_2(4k + 2, 4l), \quad k, l \in \mathbb{Z}, \\ (2k + 1, 2l + 1) & \text{if } (x, y) = (4k + 2, 4l + 2), \quad k, l \in \mathbb{Z}. \end{cases}$$

The surjection h is demonstrated in Figure 7 where, similarly to Figure 3, the corresponding decomposition of the topological space $(\mathbb{Z}^2 \times \mathbb{Z}^2, w)$ is marked by the dashed lines. All points of a class of the decomposition are mapped by h to the center point of the class given by the bold coordinates.

Proposition 2. Let D be a simple closed curve in (\mathbb{Z}^2, v) having more than four points and such that every pair of different points $z_1, z_2 \in D$ with both coordinates even satisfies $A_4(z_1) \cap A_4(z_2) \subseteq D$. Then D is a Jordan curve in (\mathbb{Z}^2, v) .

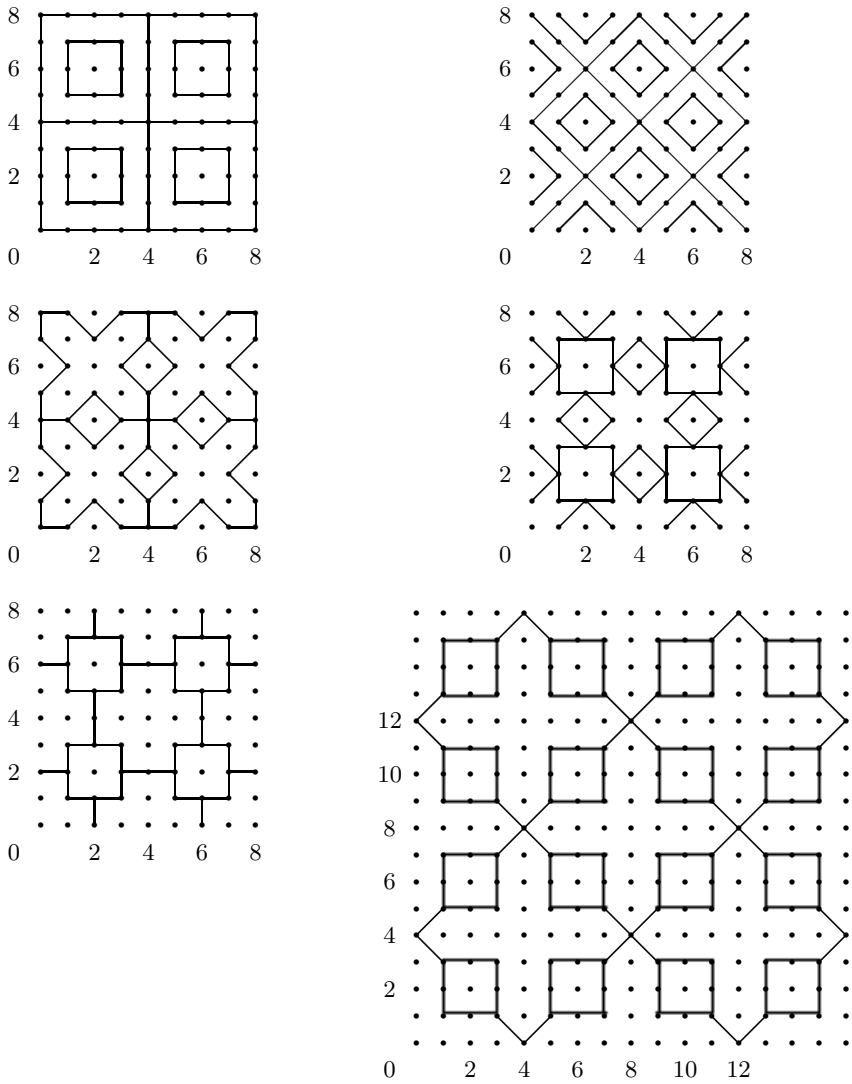


Fig. 5. Six subgraphs of the connectedness graph of w

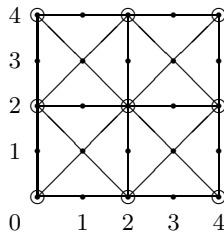


Fig. 6. The connectedness graph of v

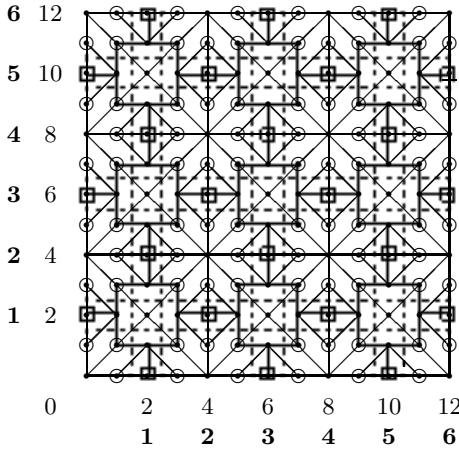


Fig. 7. Decomposition of $(\mathbb{Z} \times \mathbb{Z}, w)$ given by the surjection h

Proof. By Theorem 5, v is the quotient topology of w generated by h . It immediately follows from Theorem 4 and the definition of h that there exists a Jordan curve C in (\mathbb{Z}^2, w) such that $d(C) = D$. We may even choose C to be a cycle in the graph from Figure 3. Let C_1, C_2 be the two components of $\mathbb{Z}^2 - C$ and put $C'_i = C_i - d^{-1}(D)$ for $i = 1, 2$. Clearly, we have $C'_i \neq \emptyset$ for $i = 1, 2$. Let $(x, y) \in D$ be a point and write $d^{-1}(x, y)$ briefly instead of $d^{-1}(\{(x, y)\})$. It is evident that $d^{-1}(x, y) \subseteq C$ if and only if $x = 4k + 2$ and $y = 4l + 2$ for some $k, l \in \mathbb{Z}$ (then $d^{-1}(x, y)$ is a singleton). Thus, let $z = (x, y) \in D$ be a point with $x \neq 4k + 2$ or $y \neq 4l + 2$ for all $k, l \in \mathbb{Z}$. Then, taking into account all cases of such points, one may show that $C_i - h^{-1}(z)$ is connected for $i = 1, 2$. Consequently, C'_i is connected for $i = 1, 2$. Thus, D is a Jordan curve by Corollary 3.

4 Conclusion

It is well known that, in computer imagery, Jordan curves play an important role because they represent boundaries of regions of digital images. It is therefore useful to work with a connectedness structure on \mathbb{Z}^2 possessing certain rich enough variety of Jordan curves. We have discussed one such structure, namely the topology w . The results obtained show that the topology may be used as a background structure on \mathbb{Z}^2 to solve problems of digital image processing, especially those closely related to boundaries (image data compression, pattern recognition, boundary detection and contour filling, etc.). It provides Jordan curves that may turn, in some of its points, in the acute angle $\frac{\pi}{4}$, which is an advantage over the Khalimsky topology.

Acknowledgement. The author acknowledges partial supports from Grant Agency of the Brno University of Technology, project no. FSI-S-10-14, and from Ministry of Education of the Czech Republic, research plan no. MSM0021630518.

References

1. Čech, E.: Topological Spaces (Revised by Z.Frolík and M.Katětov). Academia, Prague (1966)
2. Eckhardt, U., Latecki, L.J.: Topologies for the digital spaces \mathbb{Z}^2 and \mathbb{Z}^3 . *Comput. Vision Image Understanding* 90, 295–312 (2003)
3. Engelking, R.: General Topology. Państwowe Wydawnictwo Naukowe, Warszawa (1977)
4. Khalimsky, E.D.: On topologies of generalized segments. *Soviet Math. Dokl.* 10, 1508–1511 (1999)
5. Khalimsky, E.D., Kopperman, R., Meyer, P.R.: Computer graphics and connected topologies on finite ordered sets. *Topology Appl.* 36, 1–17 (1990)
6. Khalimsky, E.D., Kopperman, R., Meyer, P.R.: Boundaries in digital planes. *Jour. of Appl. Math. and Stoch. Anal.* 3, 27–55 (1990)
7. Kiselman, C.O.: Digital Jordan Curve Theorems. In: Nyström, I., Sanniti di Baja, G., Borgefors, G. (eds.) DGCI 2000. LNCS, vol. 1953, pp. 46–56. Springer, Heidelberg (2000)
8. Kong, T.Y., Kopperman, R., Meyer, P.R.: A topological approach to digital topology. *Amer. Math. Monthly* 98, 902–917 (1991)
9. Kopperman, R., Meyer, P.R., Wilson, R.G.: A Jordan surface theorem for three-dimensional digital spaces. *Discr. and Comput. Geom.* 6, 155–161 (1991)
10. Marcus, D., et al.: A special topology for the integers (Problem 5712). *Amer. Math. Monthly* 77, 1119 (1970)
11. Rosenfeld, A.: Digital topology. *Amer. Math. Monthly* 86, 621–630 (1979)
12. Rosenfeld, A.: *Picture Languages*. Academic Press, New York (1979)
13. Šlapal, J.: Closure operations for digital topology. *Theor. Comp. Sci.* 305, 457–471 (2003)
14. Šlapal, J.: A digital analogue of the Jordan curve theorem. *Discr. Appl. Math.* 139, 231–251 (2004)
15. Šlapal, J.: Digital Jordan curves. *Top. Appl.* 153, 3255–3264 (2006)
16. Šlapal, J.: A quotient-universal digital topology. *Theor. Comp. Sci.* 405, 164–175 (2008)
17. Šlapal, J.: Convenient closure operators on \mathbb{Z}^2 . In: Wiederhold, P., Barneva, R.P. (eds.) IWCIA 2009. LNCS, vol. 5852, pp. 425–436. Springer, Heidelberg (2009)

Maximal Planes and Multiscale Tangential Cover of 3D Digital Objects

Emilie Charrier^{1,2} and Jacques-Olivier Lachaud¹

¹ Laboratoire de Mathématiques (LAMA), UMR 5127 CNRS
Université de Savoie, Chambéry, France

`jacques-olivier.lachaud@univ-savoie.fr`

² Grenoble Image Parole Signal Automatique (Gipsa-Lab), UMR CNRS 5216
Université Joseph Fourier, Grenoble, France

`emilie.charrier@univ-savoie.fr`

Abstract. The sequence of maximal segments (i.e. the tangential cover) along a digital boundary is an essential tool for analyzing the geometry of two-dimensional digital shapes. The purpose of this paper is to define similar primitives for three-dimensional digital shapes, i.e. maximal planes defined over their boundary. We provide for them an unambiguous geometrical definition avoiding a simple greedy characterization as previous approaches. We further develop a multiscale theory of maximal planes. We show that these primitives are representative of the geometry of the digital object at different scales, even in the presence of noise.

1 Introduction

Maximal segments [5,7] are the inextensible digital straight segments over the boundary of digital shapes. They have proven to be an essential tool for analyzing the geometry of 2D digital shapes, for instance for length and tangent estimation [14,3], curvature estimation [7,9], multiresolution analysis [6], unsupervised noise detection [10] or minimum length polygon computation. Furthermore, this theory of maximal segments can be extended to take into account possible noise in the data. The principle is simply to authorize thicker digital straight segments. This approach was proposed by Debled-Rennesson *et al.* [4], where a user specifies a maximal thickness in the segment decomposition.

Unfortunately, there is for now no equivalent theory for 3D or n D shapes, i.e. the definition and computation of maximal planes. A natural approach would be related to the convex hull, but it is not satisfactory for non convex shapes. Polyhedrization methods could also be considered as good candidates for defining maximal planes, since they are piecewise linear reconstruction of the shape boundary. Unfortunately, existing methods use greedy algorithms, whose result is for instance dependent on the starting point. Several polyhedrization methods [8,12] starts from a point and greedily aggregates surrounding points as long as they form a digital planar set. Then, it chooses arbitrarily a new point and repeat the process until all points have been visited. As the reader may check on the freely available implementation of [16], these methods do not capture well

linear or smoothly curved parts. More elaborate methods ([1] and especially [15]) address partially the problem of distinguishing polyhedral parts from smoothed parts. A user-given parameter and some *ad hoc* rules improve the previous polyhedrization algorithms. However, the obtained planes are again algorithmically obtained, without geometric or analytic description.

There exists a lot of reconstruction algorithms from scattered data in the image synthesis and computational geometry field. However, they do not take into account the specific geometry of digital spaces. For instance, if the object is the digital polyhedron, such algorithms cannot recognize digital planes as Euclidean planes. A staircase effect or an over-smoothing is generally the result of these algorithms.

The essential problem for defining interesting digital planes over a digital surface is that they must be characteristic of the local shape geometry, i.e. they should act as a local tangent plane. Now, no such problem exists in 2D, since the greedy inextensible digital linear sets — the so-called maximal segments — have been proven to be good tangent approximation [14]. However, in 3D, most inextensible digital planar sets are not characteristics of the local tangent geometry of the shape. For instance, any slice in the shape is planar, whatever the chosen direction. We must therefore find a way to select the representative planes within all these planar sets.

It is clear that the combinatorics of all planar sets of an object is too important. A natural approach is to find the smallest subset of these primitives which covers the digital boundary. But it is unlikely to find such an algorithm since this problem has been shown to be NP-complete [17].

We propose a new approach to tackle this problem by introducing the concept of *maximal (hyper)planes at a given scale* and the *hierarchy* of such primitives. Its objective is to satisfy the following requirements:

1. Maximal planes should approach the notion of tangent plane, whether the object under study is the digitization of a smooth shape with curvatures or the digitization of polyhedral surfaces.
2. It should take into account the specific nature of digital data, for instance, digital planes should be recognized in one piece.
3. Maximal planes must have a sound geometric definition. They are not only defined as the result of an algorithm, nor depend on user-given parameter(s).
4. They should be defined at different scales, so as to analyze the shape geometry at progressive resolutions. In this way, geometric analysis of noisy shapes can be addressed.
5. Ideally, this definition should encompass the classical 2D definition of maximal segments.

The paper is thus organized as follows. Maximal (hyper)planes are defined in Section 2, and their hierarchy is also presented. Section 3 proposes algorithms to compute them, a time complexity analysis is also carried out. Section 4 presents a natural application of hierarchical maximal planes: the estimation of the normal vector field of digital shapes. Results on known shapes show that maximal planes

are characteristic of the shape geometry at different scales, whether the shape is the digitization of a polyhedral or smoothly curved object. Furthermore, the presence of noise is addressed by the multiscale hierarchy. Section 5 concludes and lists several future research perspectives.

2 Definition of Maximal Hyperplanes

In this section, we define the hierarchy of maximal planes that covers any discrete surface. We restrict our study to digital surfaces of \mathbb{Z}^n , but the framework remains valid for finite subsets of \mathbb{R}^n with connectivity relations like triangulated surfaces.

2.1 Digital Surface and Tightest Hyperplane

We recall that in the 3D cell complex approach of Kovalevsky (e.g., see [13,11]), 3-cells (open unit cubes), 2-cells (open unit squares), 1-cells (open unit segments) and 0-cells (closed points) are respectively called voxels, surfels, lineles and pointels.

The surface under study is always denoted by \mathcal{S} and it corresponds to some graph (V, E) , where the set of *vertices* V is a subset of \mathbb{Z}^n and the set of *edges* E is the connectivity relation between these vertices. Note that this framework covers several definitions of digital surfaces, for instance:

- The set V may represent the border voxels of a digital object and E is then the neighborhood graph of V , choosing for instance the $(2n, 3^n - 1)$ adjacencies [Rosenfeld]
- The set V may represent the surfels centroid of a digital object boundary and E is then any bel adjacency [Herman93,Udupa94]
- The set V may represent the pointels of a digital object boundary and E is simply the grid 1-cells between them.

A subset X of \mathbb{Z}^n is called a *piece of digital hyperplane* if it corresponds to the discretization of a piece of Euclidean hyperplane. As a Euclidean hyperplane is characterized by its normal vector, the same applies for digital hyperplanes.

We usually define arithmetically digital hyperplanes as follows. The set of points X in \mathbb{Z}^n corresponds to a piece of *digital naive hyperplane* of normal vector $N = (N_1, \dots, N_n) \in \mathbb{Z}^n$ if any point p of X satisfies:

$$\mu \leq N \cdot p < \mu + \|N\|_\infty \tag{1}$$

where N is in its lowest terms. Let e_k denote the axis such that $\|N\|_\infty = N_k$, then e_k is called the *major axis* of the piece of digital hyperplane.

We can rewrite the preceding double inequality as follows :

$$\frac{\max_{p \in X}(N \cdot p) - \min_{p \in X}(N \cdot p)}{\|N\|_\infty} < 1 \tag{2}$$

We can geometrically interpret this definition of digital naive hyperplane. Consider two Euclidean hyperplanes of normal vector N such that the set of points X is contained between the two hyperplanes and such that at least one point of X lies on each hyperplane. These two hyperplanes are called *supporting hyperplanes* and (2) means that the Euclidean distance between these two hyperplanes relative to the major axis is strictly less than 1. We notice that, by allowing a larger distance between supporting hyperplanes, we can define thicker digital hyperplanes.

Obviously, a given subset X corresponds to the discretization of several Euclidean hyperplanes, resulting of a small variation of the normal vector. As a consequence, we choose the following definition, which has the advantage of defining without ambiguity one plane for a given subset X of \mathbb{Z}^n . Among all *arithmetic digital hyperplanes* which contain X , the one with the smallest axis-aligned thickness is called the *tightest hyperplane* containing X , and is written $TH(X)$. If several axes induce a tightest hyperplane, we choose the first one. The *normal vector* $N(X)$ to X is the normal vector to the tightest hyperplane, pointing in the same half-space as its axis. The *thickness of X* is the thickness of $TH(X)$, otherwise said the quantity $t(X) \stackrel{def}{=} \frac{\max_{p \in X} (N(X) \cdot p) - \min_{p \in X} (N(X) \cdot p)}{\|N(X)\|_\infty}$.

2.2 Neighborhood, ν -Thick Disk and Extension

Let $\|\cdot\|$ be the Euclidean norm. The closed ball of radius r centered at some point p is denoted by $B_p(r)$. For a given vertex p of \mathcal{S} , the set of vertices of \mathcal{S} lying in $B_p(r)$ defines a subgraph of \mathcal{S} . There may be several connected component in this subgraph, but the only one containing p is called the *r -neighborhood* of p in \mathcal{S} , and is written $\mathcal{S}_p(r)$.

The r -neighborhood of p admits a tightest plane. Its thickness is clearly an increasing function of r , denoted by w_p . Since we are considering a finite graph, this function is piecewise constant on intervals $[r_k, r_{k+1}[$. Inversely, for a given thickness ν , the radius function $\rho_p(\nu)$ is defined as follows: $\rho_p(\nu) = r_k$ iff $\nu \in [w_p(r_k), w_p(r_{k+1})[$.

The subgraph $\mathcal{S}_p(\rho_p(\nu))$ is called the *ν -thick disk* around p , and is simply denoted with \mathcal{S}_p^ν . The *ν -tightest plane around p* is defined as $TH_p^\nu \stackrel{def}{=} TH(\mathcal{S}_p^\nu)$. Its normal direction is written as \vec{N}_p^ν .

This plane is characteristic of the tangent space at p at a given scale ν . It forms a strip in the space of axis-thickness no greater than ν , which contains p and an isotropic connected neighborhood around it of radius $\rho_p(\nu)$. Having an isotropic neighborhood is interesting when analyzing the digitization of shapes with smooth boundaries. However, when the object under study contains planar facets (e.g. polyhedra, manufactured objects), this neighborhood is not always pertinent. It is therefore interesting to consider the extension to the largest connected component including p and belonging to TH_p^ν . The *ν -thick extension* around p is defined as the subgraph of \mathcal{S} :

$$\vec{\mathcal{S}}_p^\nu \stackrel{def}{=} \{\text{The connected component of } \{\mathcal{S} \cap TH_p^\nu\} \text{ that contains } p\}.$$

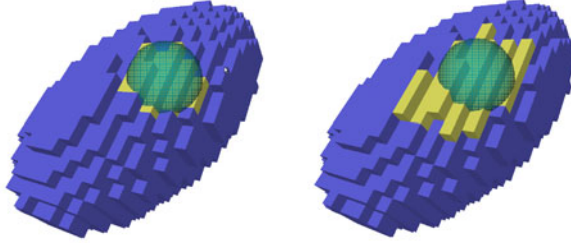


Fig. 1. A ν -thick disk and its surrounding ball (left), its ν -thick extension (right) ($\nu = 1$)

Since $\mathcal{S}_p^\nu \subset TH_p^\nu$, it is obvious that $\mathcal{S}_p^\nu \subset \overline{\mathcal{S}_p^\nu}$. The ν -thick extension thus contains the ν -thick disk around any vertex.

Figure 1 illustrates an example of a 1-thick disk (in yellow) and its surrounding ball (in green) on the surface of an ellipse (left). On the right, its 1-thick extension is represented.

2.3 Maximal Disks and Hierarchy of Active Vertices

The preceding definitions may let us think that any point of a surface induces a maximal disk and a maximal plane for a given scale. This is rather counter-intuitive. We tend to think that the coarser is the scale the simpler is the object. We would expect less maximal disks and extensions at coarser scales. We therefore introduce a mechanism to keep only the most significant disks and extensions at each scale. Furthermore, this mechanism naturally induces a hierarchy on these neighborhoods.

A ν -thick disk is *maximal* iff for any vertex $q \in \mathcal{S}$, $q \neq p$, the ball $B_p(\rho_p(\nu))$ is not included in the ball $B_q(\rho_q(\nu))$ or $p \notin TH_q^\nu$. The first condition guarantees that $\mathcal{S}_p^\nu \subset \mathcal{S}_q^\nu$ whenever p and q lie in the same component of $\mathcal{S} \cap B_p(\rho_p(\nu))$. The second condition approximates this connection condition efficiently.

We may now define a hierarchy of *active vertices* for any sequence of increasing thicknesses $(\nu_i)_{i=0, \dots, L}$, with $\nu_0 = -1$, $\nu_1 = 0$, $\nu_L = t(\mathcal{S})$. For the sake of convenience, we admit that the first thickness of the sequence is negative. The following process defines a hierarchy $(\mathcal{F}^{\nu_i})_{i=-1, \dots, L}$:

$$\mathcal{F}^{\nu_i} = \begin{cases} \{\text{set of vertices of } \mathcal{S}\} & \text{if } i = 0, \\ \{p \in \mathcal{F}^{\nu_{i-1}} : \mathcal{S}_p^{\nu_i} \text{ is maximal}\} & \text{if } i \geq 1. \end{cases} \tag{3}$$

Furthermore, if the sequence (ν_i) is refined uniformly, then the family of sets converges uniformly toward a piecewise constant family of sets $(\mathcal{F}^\nu)_{\nu \in \mathbb{R}}$, with the convenient convention that $\forall \nu < 0, \mathcal{F}^\nu = \mathcal{S}$. The sequence of values ν , such that \mathcal{F}^ν is only right-continuous, represents the *stable scales for* \mathcal{S} and are denoted as a sequence $(\mu_i)_{i=0, \dots, L'}$. By construction, this hierarchy of active vertices at progressive scales only depends on the input surface \mathcal{S} (vertices and edges).

By definition of the hierarchy $(\mathcal{F}^{\nu_i})_{i=-1,\dots,L}$, it is obvious that $\mathcal{S} = \mathcal{F}^{<0} \supset \mathcal{F}^0 \supset \dots \supset \mathcal{F}^{\mu_i} \supset \mathcal{F}^{\mu_{i+1}} \supset \dots \supset \mathcal{F}^{\mu_{L'}}$. As a consequence, we obtain the following property:

Property 1. $\forall \mu, \mu' \in \mathbb{R}, \mu < \mu' \Rightarrow \mathcal{F}^\mu \supseteq \mathcal{F}^{\mu'}$.

Moreover, we notice that at a given scale ν each vertex $q \in \mathcal{S}$ belongs to at least one ν -thick disk. This leads to the next property :

Property 2. $\forall \nu \in \mathbb{R}, \bigcup_{p \in \mathcal{F}^\nu} \mathcal{S}_p^\nu = \mathcal{S}$ and $\bigcup_{p \in \mathcal{F}^\nu} \overline{\mathcal{S}}_p^\nu = \mathcal{S}$.

Definition 1. The tangential cover \mathbb{T}^ν at scale ν of \mathcal{S} is the set of subgraphs

$$\mathbb{T}^\nu \stackrel{\text{def}}{=} \{\overline{\mathcal{S}}_p^\nu : p \in \mathcal{F}^\nu\}.$$

Each element of \mathbb{T}^ν is called a ν -maximal plane of \mathcal{S} . The set of ν -maximal planes containing a vertex p is called the ν -linear pencil of p .

Figure 2 shows an example of maximal 1-thick disk (yellow) containing two different non maximal 1-thick disk.

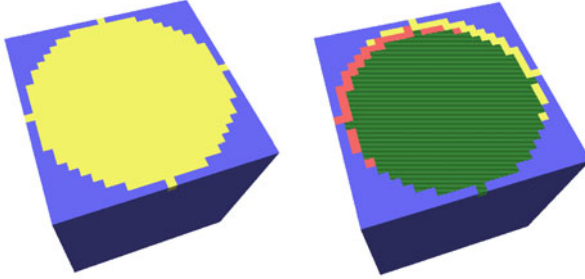


Fig. 2. 1-thick maximal disk in yellow (left), 1-thick disks included in the yellow one (right)

Figure 3 shows an example of two maximal ν -thick disks in red and yellow for $1 \leq \nu \leq 3$ (from (a) to (c)) on a noisy surface. The red one is not maximal anymore when $\nu = 4$ because it is included in the yellow one, as a consequence, its center is not an active vertex anymore at scale 4.

3 Computation and Time Complexity

In this section, we present our algorithm for the computation of the hierarchy of active vertices. This hierarchy leads to the computation of the tangential cover of the surface at different scales.

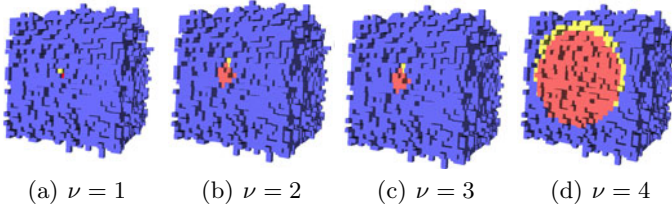


Fig. 3. Two distinct maximal ν -thick disks for $1 \leq \nu \leq 3$, inclusion of the red 4-thick disk in the maximal yellow one

3.1 Algorithm Design

Our method computes the hierarchy of active vertices $\mathcal{F}_{i=-1, \dots, K}^{\nu_i}$ for a given sequence of increasing thicknesses $(\nu_i)_{i=0, \dots, K}$, with $\nu_0 = -1, \nu_1 = 0, \dots$, relative to a given surface \mathcal{S} . It is summarized in Algorithm 1 and we return to the main steps in the following.

Algorithm 1. Hierarchy of active vertices algorithm

```

1 INPUT :  $\mathcal{S}, (\nu_i)_{i=0, \dots, K}$ 
2 OUTPUT :  $\mathcal{F}_{i=-1, \dots, K}^{\nu_i}$ 
3    $Q \leftarrow NULL$ 
4    $\mathcal{F}^{-1} \leftarrow \{\text{set of vertices of } \mathcal{S}\}$ 
5   FOR EACH increasing thickness  $\nu_i, 0 \leq i \leq K$ 
6      $L \leftarrow NULL$ 
7     FOR EACH vertex  $p$  of  $\mathcal{F}^{\nu_{i-1}}$ 
8        $Q.\text{push}(\mathcal{S}_p^{\nu_i});$ 
9     WHILE (!empty( $Q$ ))
10       $A \leftarrow Q.\text{pop}()$ 
11      IF ( $\forall B \in L, !(A \subset B)$ )
12         $L.\text{push}(A)$ 
13         $\mathcal{F}^{\nu_i} \leftarrow \mathcal{F}^{\nu_i} \cup A.\text{center}()$ 

```

By convention, \mathcal{F}^{-1} corresponds to the set of vertices of \mathcal{S} . For each thickness $\nu_i, 0 \leq i \leq K$, the set \mathcal{F}^{ν_i} is induced by the set $\mathcal{F}^{\nu_{i-1}}$ and the main steps of our approach are the following :

1. Computing the ν_i -thick disk around p , for each vertex p belonging to $\mathcal{F}^{\nu_{i-1}}$. The ν_i -thick disks are stored in a priority queue (denoted by Q) ordered relative to the radius of the disks, the largest disk lying on the top of the priority queue.
2. Extracting of the priority queue the maximal ν_i -thick disks and adding their vertex center to \mathcal{F}^{ν_i}

The first step consists in computing the ν_i -thick disk $\mathcal{S}_p^{\nu_i}$ around p , for each vertex p belonging to $\mathcal{F}^{\nu_{i-1}}$. For this, we use in an incremental way a digital plane recognition algorithm such as the COBA algorithm introduced in [2].

The method adds neighbors of p in a specific order as long as they belong to the same ν_i -thick plane. Let us call r_k -ring of a point p the subgraph defined by $\mathcal{S}_p(r_k) \setminus \mathcal{S}_p(r_{k-1})$. It adds the vertices of the r_k -ring of p , for each increasing $k > 1$. If a vertex of the r_k -ring cannot be added (the set of vertices will not correspond to a ν_i -thick disk anymore), all the vertices of r_k -ring of p are removed from the subgraph and the algorithm stops. We illustrate the construction of ν_i -thick disk around p , denoted by $\mathcal{S}_p^{\nu_i}$, in Algorithm 2. We use a priority queue, denoted by Q , which contains vertices ordered relative to their Euclidian distance to the initial point p . The closest vertex lies on the top of the queue.

Algorithm 2. The ν -thick disk algorithm

```

1 INPUT :  $\mathcal{S}, \nu, p$ 
2 OUTPUT :  $\mathcal{S}_p^\nu$ 
3    $DP \leftarrow TRUE$     $\mathcal{S}_p^\nu \leftarrow \emptyset$     $Q \leftarrow NULL$ 
4    $Q.push(p)$ 
5   WHILE(!empty( $Q$ ) AND  $DP$ )
6      $q \leftarrow Q.pop()$ 
7     IF( $q \notin \mathcal{S}_p^\nu$ )
8        $\mathcal{S}_p^\nu \leftarrow \mathcal{S}_p^\nu \cup q$ 
9       IF(isDigitalPlane( $\mathcal{S}_p^\nu, \nu$ ))
10         $Q.push(neighbors\ of\ q)$ 
11      ELSE
12         $r \leftarrow q.EuclidianDistanceTo(p)$ 
13         $\mathcal{S}_p^\nu \leftarrow \mathcal{S}_p^\nu \setminus r\text{-ring}(p)$ 
14         $DP \leftarrow FALSE$ 

```

The second step consists in extracting the maximal ν_i -thick disks and adding their vertex center to \mathcal{F}^{ν_i} . At the end of the first step, ν_i -thick disks are stored in a priority queue ordered relative to their radius. We remove each disk of the priority queue, beginning with the disk on the top, and we wonder whether it is maximal or not. If it is maximal, it is added to a queue called L and its center is added to \mathcal{F}^{ν_i} . Obviously, a disk cannot be included in one of smaller radius. We also notice that, because two disks cannot have the same center, a disk cannot be included in one of the same radius. The first extracted disk is obviously maximal and so it is added to the queue L . By construction of the priority queue, for each extracted disk d , we only have to test the inclusion of d in the disks of the queue L .

3.2 Time Complexity Analysis

We propose to analyse the time complexity of the hierarchy of active vertices algorithm (Algorithm 1).

Let \mathcal{S} denote the surface under study, let us denote by D and m its diameter and the number of vertices of its associated subgraph respectively. According to Property 1, for all i such that $0 \leq i \leq k$, \mathcal{F}^{ν_i} is included in or equal to $\mathcal{F}^{\nu_{i-1}}$. As a consequence, the number of vertices of each \mathcal{F}^{ν_i} is bounded by m , for all i such that $-1 \leq i \leq k$. As the time complexity of the computation of each \mathcal{F}^{ν_i}

only depends on the cardinality of its ascending set and of the diameter of the surface, we can study separately the time complexity of the computation of a set \mathcal{F}^{ν_i} whatever ν_i .

For a given thickness ν_i , the first step consists in computing the maximal ν_i -thick disks centered on each vertex of the set $\mathcal{F}^{\nu_{i-1}}$ and adding it to priority queue. To compute each ν_i -thick disk, the COBA algorithm is used to incrementally construct a maximal isotropic piece of digital plane of thickness ν_i . As in the non-incremental case, the COBA algorithm runs in $O(m \log(D))$ time (see [2]). The pushing operation on a priority queue runs in $O(\log(l))$ time where l denotes the size of the priority queue. As the size of our priority queue is obviously bounded by m , this step runs in $O(\log(m))$ time. Because $O(\log(m))$ can be neglected relative to $O(m \log(D))$, the first step runs in $O(m^2 \log(D))$ time.

The second step consists in extracting the maximal ν_i -thick disks. For each ν_i -thick disk of the priority queue, the method tests whether it is included in a larger one. As the number of larger ν_i -thick disks is bounded by m for each disk, deciding whether it is maximal or not runs in $O(m)$ time. As a consequence, checking the maximality of every disks of the priority queue runs in $O(m^2)$ time.

To conclude, as $O(m^2)$ can be neglected relative to $O(m^2 \log(D))$, the computation of the set \mathcal{F}^{ν_i} for a given thickness ν_i runs in $O(m^2 \log(D))$ time in the worst case. The computation of all the hierarchy of active vertices for $(\nu_i)_{i=0,\dots,L}$ runs in $O(Lm^2 \log(D))$ in the worst case.

4 Application

We proposed in Section 3 an algorithm to compute the hierarchy of active vertices at different scales. Moreover, we introduced the definition of the tangential cover of a surface \mathcal{S} at a given scale ν (see Definition 1), induced by the set of active vertices at scale ν .

Knowing the tangential cover \mathbb{T}^ν of a surface \mathcal{S} at a scale ν naturally leads to the normal estimation of the surface at each point at this scale. Indeed, each ν -maximal plane of \mathbb{T}^ν represents a tangent plane for each covered vertex. To each vertex v of the surface, we associate the average of the normal vectors of each ν -maximal plane of its ν -linear pencil.

For a regular discrete surface, we expect the normal estimation to be well representative at scale 1. Conversely, if the discrete surface is noisy, the normal estimation could be wrong for some vertices representing noise at scale 1. Nevertheless, we expect that at coarser scales our normal estimation gets improved, until we reach the stable scale. This stable scale represents the global noise level added on the surface.

We implement our method in C++ using the ImaGene library. We generate three-dimensional regular discrete shapes, we choose to add noise or not, we compute its tangential cover at a given scale and we estimate a normal vector at each surface voxel. In order to visualise the result, we display the three-dimensional surface using Inventor by associating to each surface voxel its estimated normal vector. As a consequence, when the surface is lighted, we can immediately judge whether the normal estimation is consistent.

Figure 4 shows the result of our normal estimation at scale 1 on the regular surface of a cube, of a sphere and of an ellipse. Figure 5 and figure 6 show the result of our normal estimation on noisy surfaces of a cube and of an ellipse respectively, at scales 1, 2, 3 and 4. We notice that the normal estimation gets more and more consistent until stability at scale 4.

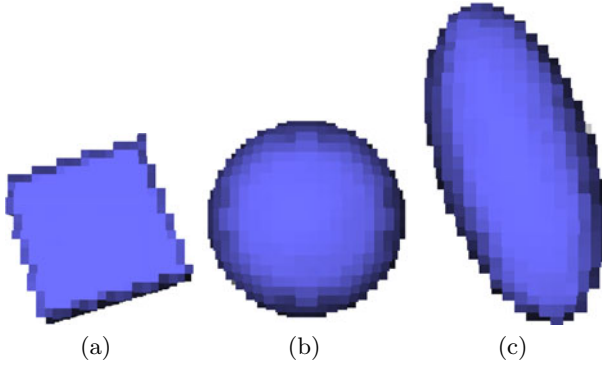


Fig. 4. Normal estimation at scale 1 of regular discrete surfaces

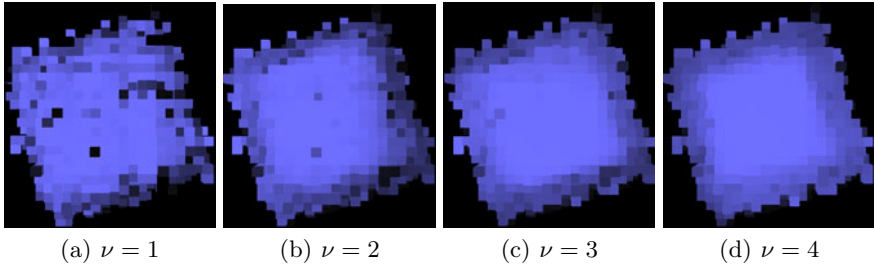


Fig. 5. Normal estimation at scale 1, 2, 3 and 4 of a noisy discrete surface : a cube

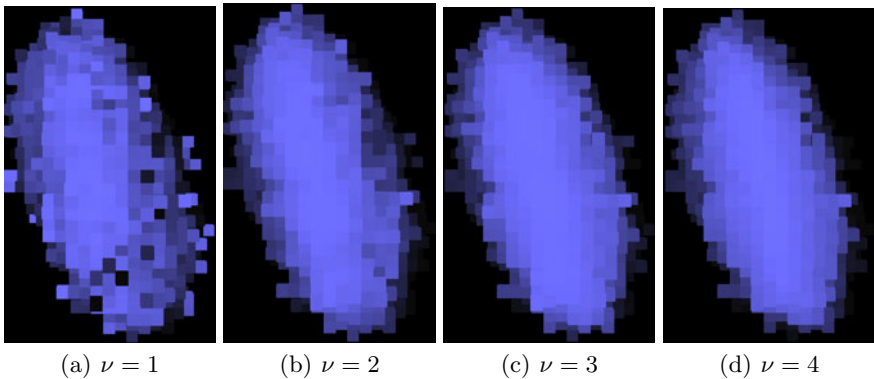


Fig. 6. Normal estimation at scale 1, 2, 3 and 4 of a noisy discrete surface : an ellipse

Finally, we show the result of our normal estimation on realistic data representing an old car (data available on the TC18 website¹). Figure 7 shows the original object and the result of our normal estimation at scale 1. Figure 8 shows the original object with additional noise and the result of our normal estimation on this noisified object at scale 1, 2 and 3.

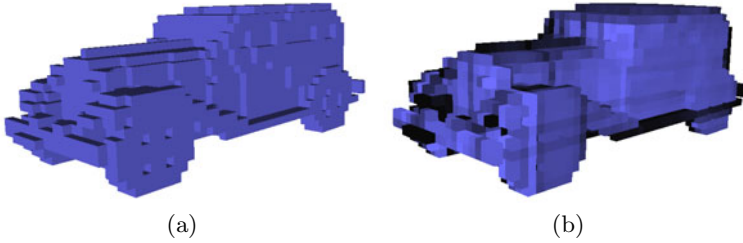


Fig. 7. Discrete object representing a Dodge (left), normal estimation at scale 1 (right)

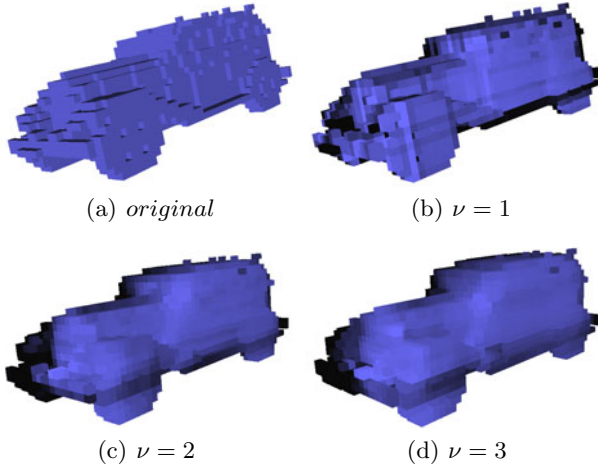


Fig. 8. Original object and normal estimation at scale 1, 2 and 3 of a noisy discrete object : a dodge

5 Conclusion and Perspectives

We propose in this paper a new definition for the tangential cover of a three-dimensional digital shape based on the computation of maximal planes over its surface. Moreover, we provide an algorithm to compute the hierarchy of active vertices at different scales, which induces the tangential cover of the shape at each scale. We highlight the fact that maximal planes are locally representative of the shape and so that they can be used to estimate the normal vector at each point on the surface. The study of maximal planes over a discrete surface could also

¹ <http://tc18.liris.cnrs.fr>

lead to estimation of local or global noise level on a three-dimensional surface. Moreover, we think that, as they provide an estimation of the normal vector at each point, they could also be used to extract other geometric characteristics as the curvature. We could work on them to discriminate curve parts from flat parts on the surface or to divide up the surface in convex and in concave parts.

References

1. Burguet, J., Malgouyres, R.: Strong thinning and polyhedral approximation of the surface of a voxel object. *Discrete Applied Mathematics* 125(1), 93–114 (2003)
2. Charrier, E., Buzer, L.: An efficient and quasi linear worst-case time algorithm for digital plane recognition. In: Coeurjolly, D., Sivignon, I., Tougne, L., Dupont, F. (eds.) DGCI 2008. LNCS, vol. 4992, pp. 346–357. Springer, Heidelberg (2008)
3. de Vieilleville, F., Lachaud, J.-O.: Comparison and improvement of tangent estimators on digital curves. *Pattern Recognition* 42(8), 1693–1707 (2009)
4. Debled-Rennesson, I., Feschet, F., Rouyer-Degli, J.: Optimal blurred segments decomposition of noisy shapes in linear times. *Computers and Graphics* 30, 30–36 (2006)
5. Dorst, L., Smeulders, A.W.M.: Decomposition of discrete curves into piecewise straight segments in linear time. In: *Vision Geometry. Contemporary Mathematics*, vol. 119, pp. 169–195 (1991)
6. Feschet, F.: Multiscale analysis from 1d parametric geometric decomposition of shapes. In: *Proc. ICPR*, pp. 2102–2105 (2010)
7. 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)
8. Françon, J., Papier, L.: Polyhedrization of the boundary of a voxel object. In: Bertrand, G., Couprie, M., Perroton, L. (eds.) DGCI 1999. LNCS, vol. 1568, pp. 425–434. Springer, Heidelberg (1999)
9. Kerautret, B., Lachaud, J.-O.: Curvature estimation along noisy digital contours by approximate global optimization. *Pattern Recognition* 42(10), 2265–2278 (2009)
10. Kerautret, B., Lachaud, J.-O.: Multi-scale analysis of discrete contours for unsupervised noise detection. In: Wiederhold, P., Barneva, R.P. (eds.) IWCIA 2009. LNCS, vol. 5852, pp. 187–200. Springer, Heidelberg (2009)
11. Klette, R., Rosenfeld, A.: *Digital Geometry: Geometric Methods for Digital Picture Analysis*. Morgan Kaufmann Publishers Inc., San Francisco (2004)
12. Klette, R., Sun, H.J.: Digital planar segment based polyhedrization for surface area estimation. In: Arcelli, C., Cordella, L., di Baja, G.S. (eds.) IWVF 2001. LNCS, vol. 2059, pp. 356–366. Springer, Heidelberg (2001)
13. Kovalevsky, V.A.: Finite topology as applied to image analysis. *Comput. Vision Graph. Image Process.* 46, 141–161 (1989)
14. Lachaud, J.-O., Vialard, A., de Vieilleville, F.: Fast, accurate and convergent tangent estimation on digital contours. *Image and Vision Computing* 25(10), 1572–1587 (2007)
15. Provot, L., Debled-Rennesson, I.: 3D noisy discrete objects: Segmentation and application to smoothing. *Pattern Recognition* 42(8), 1626–1636 (2009)
16. Sivignon, I.: Digital surface decomposition (dsd) into digital plane segments (2006), <http://liris.cnrs.fr/isabelle.sivignon/DSD.html>
17. Sivignon, I., Coeurjolly, D.: Minimum decomposition of a digital surface into digital plane segments is np-hard. *Discrete Applied Mathematics* 157(3), 558–570 (2009)

Recognition of Digital Hyperplanes and Level Layers with Forbidden Points

Laurent Provot and Yan Gerard

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

Abstract. We consider a new problem of recognition of digital primitives – digital hyperplanes or level layers – arising in a new practical application of surface segmentation. Such problems are usually driven by a maximal thickness criterion which is not satisfactory for applications as soon as the dimension of the primitives becomes greater than 1. It is a good reason to introduce a more flexible approach where the set to recognize (whose points are called inliers) is given along with two other sets of outliers that should each remain on his own side of the primitive. We reduce this problem of recognition with outliers to the separation of three point clouds of \mathbb{R}^d by two parallel hyperplanes and we provide a geometrical algorithm derived from the well-known GJK algorithm to solve the problem.

Keywords: Digital Hyperplane Recognition, Outliers, Linear Programming, GJK.

1 Problem Statement

1.1 Why a New Problem of Recognition?

The starting point of this paper is an application of digital shape decomposition into algebraic primitives. Since 15 years, many researches have provided a large literature about the decomposition of two- or three-dimensional digital shapes – a subset of \mathbb{Z}^2 or \mathbb{Z}^3 – in digital straight segments or pieces of planes [6,12,11]. There is however now a large interest for working with digital algebraic primitives of degree greater than 1, such as digital circles, parabola, conics and more generally with digital algebraic manifolds of any degree. Although the degree becomes greater than 1, the problem remains linear after the construction of a new geometrical instance based on quadratic or higher degree structures. Hence, the recognition of such nonlinear objects can for instance be encoded as a problem of digital hyperplane recognition. Many efficient algorithms have been proposed to solve this classical problem of Digital Geometry, from the linear time algorithm to recognize digital straight segments [6], to Computational Geometry [7,4] or Linear Programming approaches [3,11,2]. Some variations around these recognition problems have also been investigated more recently, for instance with the problem of optimization of the best consensus lines and planes [13].

This reduction from algebraic manifolds fitting to digital hyperplanes recognition presents nevertheless an important drawback. Let us explain where is the challenge by considering for instance the digital conic recognition problem.

Let S be a 8-connected curve in \mathbb{Z}^2 and C denote a conic. We consider a family of increasing conic strips C_δ around C depending on a maximal distance $\delta \geq 0$ (C_0 is exactly C). Whatever the exact definition of C_δ (several choices are possible), the recognition problem of a digital conic is given as follows (Fig. 1):

Problem 1. INPUT: S, δ_{max} .

OUTPUT: Existence of a conic C such that S is a subset of $C_{\delta_{max}}$.

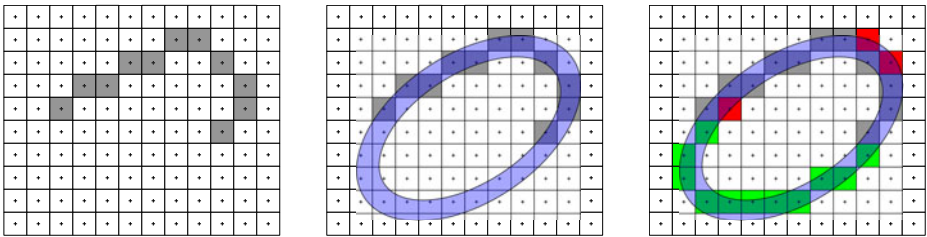


Fig. 1. S is a 8-connected curve (on the left). The recognition problem consists in finding a conic C such that S is in C_δ (a satisfying strip C_δ is drawn in the middle). There is however no guarantee – as there exists with linear objects – that the neighbors on both sides of the curve do not belong to C_δ . The problem with this approach, if we consider now the digitization of C as the trace of C_δ on \mathbb{Z}^2 , is that there are extra-points (in red) which can be strongly undesired.

Even if we admit that we have an algorithm to compute a solution, at least one question remains: How should δ_{max} be chosen? This choice is easy in the framework of linear structures or circles, because for some values of δ_{max} , we have arithmetical properties that lead to choose them, but it is no more true in the framework of higher degree structures. Hence, the question about the choice of δ_{max} cannot be easily derived from the properties of the lattice points in C_δ .

We nevertheless claim that by choosing δ_{max} first and searching C in a second time is not necessarily the best strategy for applications. There is an important problem of reversibility because the conic strip $C_{\delta_{max}}$ can contain many other points than the ones of S . In some regions far from the points of S , it is not a problem: we can consider that these points of $(C_{\delta_{max}} \setminus S) \cap \mathbb{Z}^2$ are the extension of S in these regions. It becomes a problem if $(C_{\delta_{max}} \setminus S) \cap \mathbb{Z}^2$ contains neighbors of S that cannot be considered as the extension of the digital curve. A good choice of δ_{max} avoids this problem in the case of linear structures but it is no more true in the general framework. The conic $C_{\delta_{max}}$ can have bad configurations. Hence, there is a risk that its trace on \mathbb{Z}^2 adds undesired neighbors to S (Fig. 1). Having these undesired neighbors in the digitization of the conic C and losing the property of local reversibility is a bad point. It is however not unavoidable, but we have to consider the recognition problem from another point of view. We think that instead of choosing δ_{max} according to a criterion that remains to

determine, it would be better to provide directly in the input the points of the curve of S and the points that we do not want to be added in a digitization of S , namely a set of points called *inliers* and another set of points called *outliers* (Fig. 2). It leads to the following formulation of the recognition problem:

Problem 2. INPUT: S_{in} and S_{out} (the undesired neighbors).
 OUTPUT: Existence of a conic C and a positive real δ such that S_{in} is a subset of C_δ and such that $S_{out} \cap C_\delta$ is empty.

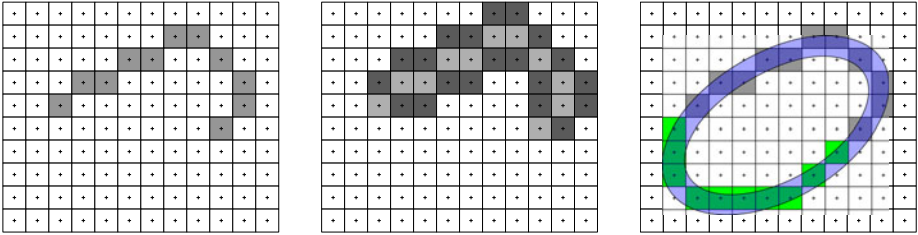


Fig. 2. S_{in} is a 8-connected curve drawn on the left. Its points are called *inliers*. We can choose or compute automatically a second set S_{out} of points called *outliers* that should not be added in a digitization of S_{in} . The inliers and outliers are shown in the middle. In this framework, the recognition problem is to find a conic strip C_δ that contains all the inliers and no outlier. The value δ is not given *a priori*. It is replaced by the outliers in the input. A solution is drawn on the right.

The value δ is not fixed *a priori*. The undesired neighbors are given in the input instead and we think that it is more suitable for the applications. Procedures to choose them automatically are easy to obtain, for example by choosing as undesired neighbors all the points of the complement of S (i.e. $\mathbb{Z}^2 \setminus S$) which have at least two 8-neighbors in S .

This approach has been used for instance in [5], in the framework of digital arc recognition, and we suggest to generalize it in this paper.

1.2 Exact Formulation

Now that we have precised the kind of the recognition problems we consider – with inliers, outliers and not with a fixed maximal thickness – it remains to set an important parameter of the problem: How do we define the continuous strip L_δ around a continuous curve or surface L of given equation $f(x) = 0$?

– Mathematical morphology suggests to define L_δ as the Minkowski sum of L with a structuring element which is usually a ball of radius δ or $\frac{\delta}{2}$ for a given norm ($\|\cdot\|_\infty$, $\|\cdot\|_2$ or $\|\cdot\|_1$). But in this framework, even with degree 2, the Problem 1 of recognition presents combinatorial difficulties which are not yet solved and the difficulties will of course increase in 3D and with degrees greater than 2. The introduction of outliers instead of δ_{max} does not simplify the question.

– A second option is to define L_δ by a double-inequality $-\frac{\delta}{2} \leq f(x) \leq \frac{\delta}{2}$. Such strips, between two level sets of equations $f(x) = -\frac{\delta}{2}$ and $f(x) = \frac{\delta}{2}$, are

called Level Layers and Digital Level Layers [8] (DLL for short) if we consider lattice sets. In the framework of DLL, Problem 1 can be easily solved by Linear Programming or Computational Geometry. But what about Problem 2? Let us rewrite it in terms of DLL characteristics:

Problem 3. INPUT: Two subsets S_{in} and S_{out} of \mathbb{Z}^d , a linear space of functions F generated by n functions f_1, f_2, \dots, f_n from \mathbb{Z}^d to \mathbb{R} .
 OUTPUT: Existence of a strip L_δ of double inequality $-\frac{\delta}{2} \leq f(x) \leq \frac{\delta}{2}$ such that $S_{in} \subset L_\delta$, the intersection $L_\delta \cap S_{out}$ is empty, the function f is in F and δ is a positive real.

It follows that, given a solution L_δ , some outliers are on one side, i.e. $f(x) < -\frac{\delta}{2}$, while other outliers are on the other side, i.e. $\frac{\delta}{2} < f(x)$. There are two things that we can notice:

- In practice the partition of the outliers into two sets on both sides of L_δ is usually easy to obtain before the resolution of Problem 3.
- Even with a linear space of functions F which are just affine combination of the coordinates, Problem 3 remains difficult. The missing information to make it easier, from a computational point of view, is the partition of the outliers according to the side they are from the strip L_δ . Hence, we have an information which is easy to obtain in practice and which would be of much importance to reduce the computational complexity of our problem. It leads to consider a variant of Problem 3 where this missing information is added in the input.

Problem 4. INPUT: A first subset $S_{in} \subset \mathbb{Z}^d$ of inliers and two subsets $S_{down}, S_{up} \subset \mathbb{Z}^d$ of outliers, a linear space of function F generated by n functions f_1, f_2, \dots, f_n from \mathbb{Z}^d to \mathbb{R} .
 OUTPUT: Existence of a strip L_δ of double inequality $-\frac{\delta}{2} \leq f(x) \leq \frac{\delta}{2}$ such that $S_{in} \subset L_\delta$, with $f(x) < -\frac{\delta}{2}$ for all outliers of S_{down} and $\frac{\delta}{2} < f(x)$ for all outliers of S_{up} , where the function f is in F and δ is a positive real.

The purpose of the paper is to provide an efficient method to solve Problem 4. This method is a variant of the well-known GJK algorithm used in order to compute the distance between the convex hulls of two point clouds in \mathbb{R}^d [9].

The next section is devoted to the reduction of the general case of Problem 4 with arbitrary functions f_i to a problem of affine Computational Geometry. In Section 3, we present our variant of GJK, while in Section 4 the experimental results and the potential applications of such a technique are presented.

2 From Arbitrary Functions to Affine Separation

2.1 Computation by Linear Programming

Let us start with the investigation of the properties of Problem 4. There are two unknowns: the real value δ and the function f in the finitely generated space F . Let us denote by a_i its coefficients: $f = \sum_{i=1}^n a_i f_i$. Hence, the problem is to find the n real coefficients a_i and δ verifying:

- (i) $\sum_{i=1}^n a_i f_i(x) < -\frac{\delta}{2}$ for all outliers of S_{down} ,
- (ii) $-\frac{\delta}{2} \leq \sum_{i=1}^n a_i f_i(x) \leq \frac{\delta}{2}$ for all inliers, i.e. all the points of S_{in} ,
- (iii) $\frac{\delta}{2} < \sum_{i=1}^n a_i f_i(x)$ for all outliers of S_{up} .

All these constraints are linear in the unknowns a_i and δ . It follows that Problem 4 can be easily solved by linear programming. We can even notice that the problem is homogeneous: if $(\delta, (a_i)_{1 \leq i \leq n})$ is a solution, then for any positive real λ , $\lambda(\delta, (a_i)_{1 \leq i \leq n})$ is also a solution. Hence, the first idea, which is to choose the minimization of δ as objective function, is not interesting. Anyway, Linear Programming allows us to solve the problem. If we assume that the dimension n of F is fixed, Megiddo algorithm even provides a linear worst case complexity in $O(|S_{in}| + |S_{down}| + |S_{up}|)$ where $|S_{in}| + |S_{down}| + |S_{up}|$ is the sum of the cardinalities of the input sets [10].

2.2 Rewriting the Problem in Terms of Computational Geometry

We assume now that one of the functions f_i is a constant, say, without loss of generality, $f_n = -1$. It follows that the equation of the level set $f(x) = 0$ (where the constant term is hidden in f) can be rewritten $\sum_{i=1}^{n-1} a_i f_i(x) = a_n$. It means that a_n is just playing the role of the constant. This assumption is completely consistent with the practice: level sets are of the form $g(x) = cste$ with g in the linear space G . This equation can be rewritten $f(x) = 0$ with f in F only if F is in the sum of the constants and G . It is in particular the case for the algebraic curves and surfaces: there is of course a constant in the generator.

We introduce now the notations F , h and h' .

- Let F be the function from $F : \mathbb{Z}^d \rightarrow \mathbb{R}^{n-1}$ defined by $F(x) = (f_i(x))_{1 \leq i \leq n-1}$.
- Let h and h' be $h = a_n - \frac{\delta}{2}$ and $h' = a_n + \frac{\delta}{2}$. These two new variables replace in fact δ and a_n . It follows that the linear constraints (i), (ii) and (iii) of Problem 4 can simply be rewritten respectively $\sum_{i=1}^{n-1} a_i f_i(x) < h$ for all outliers $x \in S_{down}$, $h \leq \sum_{i=1}^{n-1} a_i f_i(x) \leq h'$ for all the inliers and $h' < \sum_{i=1}^{n-1} a_i f_i(x)$ for all outliers of S_{up} .
- For convenience, we denote a the vector composed of the $n - 1$ coefficients a_i , i.e. $a = (a_i)_{1 \leq i \leq n-1}$.

We can eventually replace the sum $\sum_{i=1}^{n-1} a_i f_i(x)$ in the inequalities (i), (ii) and (iii) by the dot product $a \cdot F(x)$. It provides a simplified expression of Problem 4. Find a , h and h' verifying $a \cdot y < h$ for all $y \in F(S_{down})$, $h \leq a \cdot y < h'$ for all $y \in F(S_{in})$ and $h' < a \cdot y$ for all $y \in F(S_{up})$.

We can notice that if we have a solution in the opposite direction ($a \cdot y < h$ for all $y \in F(S_{up})$ and $h' < a \cdot y$ for all $y \in F(S_{down})$) we just have to take $-a$ as the new a , $-h$ as the new h' and $-h'$ as the new h to reorder the inequality in the requested direction. Hence, the sense of the inequalities is of no importance in the problem. We can notice at last that $h \leq a \cdot y < h'$ is the equation of an affine strip of \mathbb{R}^{n-1} while $a \cdot y < h$ and $h' < a \cdot y$ characterize the two half-spaces on both sides of the strip. Thus, Problem 4 has been rewritten into a geometrical problem in \mathbb{R}^{d-1} :

Problem 5. INPUT: A first subset $Q_{in} \subset \mathbb{R}^{n-1}$ of inliers and two subsets $Q_{down}, Q_{up} \subset \mathbb{R}^{n-1}$ of outliers.

OUTPUT: Existence of an affine strip $L_{a,h,h'}$ of double inequality $h \leq a.x \leq h'$ with Q_{in} in $L_{a,h,h'}$, Q_{down} on one side and Q_{up} on the other side.

Under the assumption that $f_n = -1$, we have reduced Problem 4 to Problem 5 by taking $Q_{in} = F(S_{in})$, $Q_{down} = F(S_{down})$ and $Q_{up} = F(S_{up})$ as input sets. The solutions a, h and h' of Problem 5 provide the solutions of Problem 4 with $a = (a_i)_{1 \leq i \leq n-1}$ for the indices i from 1 to $n - 1$, $a_n = \frac{h+h'}{2}$ and $\delta = h' - h$.

The geometrical formulation of the Problem 5 is easy to understand. The input provides three sets and the problem is to separate them by two parallel hyperplanes (Q_{in} in the middle, Q_{down} and Q_{up} on both sides). It is a problem of affine separation of three sets while the classical separation problem usually arises in Computational Geometry with only two sets.

3 The GJK Algorithm and Our Variant

The GJK distance algorithm 9 is an iterative method for computing the minimal distance between the convex hulls of two point clouds in arbitrary dimension. In 3D, this technique is used to detect collisions between 3D objects in real-time applications such as video games. This algorithm can be used in order to separate two point clouds S_1 and S_2 because the closest pair of points in their convex hulls – the first point is in $ConvexHull(S_1)$, the other in $ConvexHull(S_2)$ – corresponds exactly, by duality, to the largest affine strip which separates S_1 from S_2 . If the distance is null, then the two point clouds cannot be separated.

The aim of this section is to explain our variant of GJK which allows us to separate, if it is possible, three sets S_{in}, S_{up} and S_{down} by two parallel hyperplanes. It requires however to understand how the classical version of the algorithm works. We refer of course to the initial paper 9 for complete explanations, but we provide in the next subsection the main ideas.

3.1 The Classical GJK Algorithm

Let us consider two point clouds S_1 and S_2 of \mathbb{R}^d . We can denote by $x_1 \in ConvexHull(S_1)$ and $x_2 \in ConvexHull(S_2)$ the pair of points (or in degenerated cases, one of the pairs) that provide the minimal distance between the convex hulls of S_1 and S_2 . The first idea of GJK is to consider the difference body obtained by taking the convex hull of $S = S_2 - S_1 = \{y_2 - y_1 \mid y_1 \in S_1, y_2 \in S_2\}$. By construction, the difference $x_2 - x_1$ is the closest point to the origin in the convex hull $ConvexHull(S_2 - S_1)$.

It means that we can reduce in some way the initial problem to the particular case where one set is S and the other is reduced to the origin. In this framework, the idea is to work at each step with a current simplex CS whose vertices belong to S and whose dimension can increase and decrease all along the algorithm. In fact we start the algorithm with an initial simplex of dimension 0 (a point) and

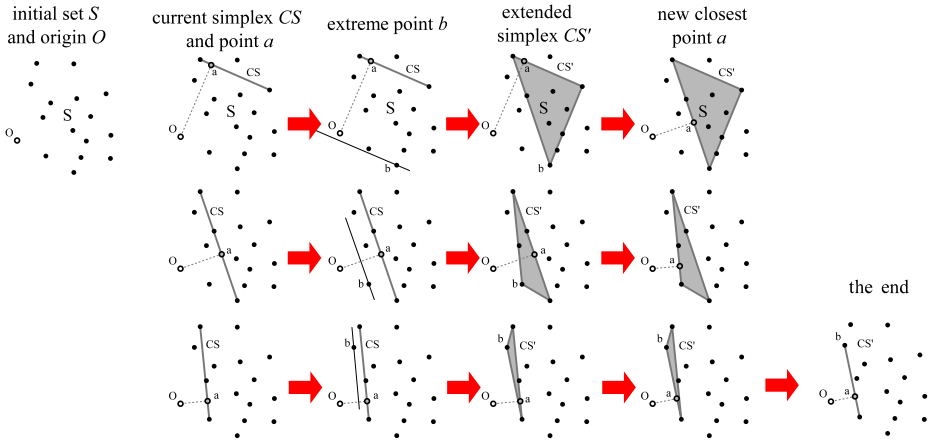


Fig. 3. On the left, the initial set S and the origin O . We want to compute the distance of $\text{ConvexHull}(S)$ to the origin. In the first column, we have a current simplex CS whose vertices belong to S , and its closest point to the origin a . In the second column, we compute the extreme point b of S according to the direction a . In the third column, we add the vertex b to the simplex CS which yields the simplex CS' . In the fourth column, we compute the new point a of the simplex CS' which is the closest to the origin. We go one row down (and to the left) by taking the face of CS' which contains a as the new current simplex CS . On the bottom right, we reach the **Case 1** which ends the computation and provides the closest point a to the origin.

the algorithm guarantees that the dimension is always less than or equal to d . Let us denote a the closest point of CS to the origin. Two cases can arise:

- **Case 1:** All the points of S (and thus all the points of the convex hull of S) verify $a \cdot x \geq a \cdot a$. It follows that a is the closest point to the origin. It ends the computation.
- **Case 2:** The minimum of $a \cdot x$ among all the points x in S is strictly less than $a \cdot a$. We introduce a point b of S providing a minimum. We extend the current simplex CS into CS' by adding the vertex b . By construction, the distance between CS' and the origin is strictly less than the distance with CS . We compute the new closest point to the origin in CS' and update a . There are two sub-cases:
 - **Case 2.1:** The point a is in the interior of CS' : it is only possible if CS' contains the origin, which means that we are in a non-separable case (convex hulls overlap).
 - **Case 2.2:** Otherwise a is on a face of CS' . It follows that there are unnecessary vertices in CS' . We remove them by taking as the new current simplex CS the face of smallest degree of CS' containing a . And we start again until the case 1 or 2.1 occurs (Fig. 3).

We should notice that the computation of b is done directly on S_1 and S_2 – in linear time wrt. the cardinality of these two sets. Last, we have the guarantee that there is no loop because the norm of a decreases at each step.

3.2 Our GJK Variant

We have now three input sets S_{down} , S_{in} and S_{out} . It is possible to separate them by two parallel hyperplanes if and only if it is possible to separate the origin from the union $(S_{up} - S_{in}) \cup (S_{in} - S_{down})$. It means that instead of working with $S = S_2 - S_1$ as in the classical version of the algorithm, we just have to work with the set $S = (S_{up} - S_{in}) \cup (S_{in} - S_{down})$.

Algorithm 1. GJK'nD

Data: Three point sets S_{in} , S_{up} and S_{down} in \mathbb{Z}^d .

Result: The existence of an affine strip $L_{a,h,h'}$ that separates S_{in} from S_{up} and S_{down} .

```

1 begin
  // Initialization
2    $P_{in} \leftarrow \text{pickPoint}(S_{in})$ 
3    $P_{up} \leftarrow \text{pickPoint}(S_{up})$ 
4    $CS \leftarrow P_{up} - P_{in}$ 
5    $\mathbf{a} \leftarrow P_{up} - P_{in}$ 
  // Main loop
6   do
7      $b \leftarrow \text{supportPoint}(\mathbf{a}, S_{in}, S_{up}, S_{down})$ 
8     if  $\mathbf{a} \cdot b = \mathbf{a} \cdot \text{pickPoint}(CS)$  then
9       break
10     $CS' \leftarrow \text{addVertex}(b, CS)$ 
11     $\mathbf{a} \leftarrow \text{closestPointToOrigin}(CS')$ 
12     $CS \leftarrow \text{removeVertices}(CS', \mathbf{a})$ 
13  if  $\text{dimension}(CS) = d$  then
14    return false
15  else
16    //  $h \leftarrow -\mathbf{v} \cdot \text{supportPoint}(-\mathbf{v}, S_{in})$ ,
    //  $h' \leftarrow \mathbf{v} \cdot \text{supportPoint}(\mathbf{v}, S_{in})$ 
    return true

```

The algorithm [1](#) summarizes the main part of our GJK variant. The steps are actually very similar to the classical GJK algorithm except that some functions have been adapted to deal with the third point set. Let us detail these different steps.

First we initialize the closest simplex to the origin (CS) with an arbitrary point in $S_{up} - S_{in}$ (we could also choose a point in $S_{in} - S_{down}$, it does not really matter). We then start the main loop to incrementally converge toward the real closest simplex to the origin.

The function `addVertex(b, CS)` add the vertex b to the simplex CS . The function `closestPointToOrigin(CS')` returns the closest point a to the origin in the simplex CS' while function `removeVertices(CS', \mathbf{a})` returns the minimal face of the simplex CS' containing the point a . These two functions are of course deeply related. For that purpose, we use a recursive decomposition of the simplex

CS' into its faces. Let us suppose that the simplex K of dimension k is a face of CS' . We compute the barycentric coordinates of the orthogonal projection of the origin wrt. this k -simplex. If all the barycentric coordinates $\lambda_{i(1 \leq i \leq k+1)}$ are strictly positives, it means the k -simplex is the closest to the origin. If it is not the case, we go one step further in the recursion and check all its $k-1$ -simplices that are associated to a negative λ_i coordinate⁴. We can note that this technique is however not optimal because a k -simplex in \mathbb{R}^d belongs to $d-k$ simplices of higher degree so it might be checked several times. It would be worth making this an iterative procedure. But the recursive procedure has been coded and we give some results in the next section.

The most time consuming function for large instances is `supportPoint`. It computes at each step the “furthest” point b of S wrt. the direction \mathbf{a} . It is the main difference with the classical GJK algorithm as we have to deal with three sets. To do so, we do not need to explicitly compute the set S . We can indeed notice that for the set $S_{up} - S_{in}$ the point b that maximizes the value $\mathbf{a}.b$ is given by $P = M_{up} - m_{in}$ where M_{up} and m_{in} are the points that respectively maximize the value $\mathbf{a}.x$ for all $x \in S_{up}$ and minimize the value $\mathbf{a}.x$ for all $x \in S_{in}$. The same holds for $S_{in} - S_{down}$ with $Q = M_{in} - m_{down}$. To have the desired b we just have to keep between P and Q the point associated to the $\min(\mathbf{a}.P, \mathbf{a}.Q)$. The computations are thus linear wrt. the cardinality of S_{in} , S_{up} and S_{down} .

The next step is to check the validity of b . If $\mathbf{a}.b = \mathbf{a}.pickPoint(CS)$ it means CS was the closest simplex to the origin so we can stop the algorithm.

4 Experimental Results and Potential Applications

4.1 Experimental Results

To show the effectiveness of our approach, the GJK'nd algorithm has been coded and tested over different point sets lying in various dimensions. The input of the algorithm is three point sets in \mathbb{Z}^d . We have thus studied the behaviour of the algorithm according to k , the number of points of the sets, and the dimension d . The experiments have been carried out on both separable and non separable point sets. To generate those point sets in dimension d , a normal vector n is picked out randomly and the characteristics of a thick digital hyperplane (between $10\|n\|_1$ and $100\|n\|_1$) are computed. The lower and upper bounds are chosen such that $h = -\frac{\delta}{2}$ and $h' = \frac{\delta}{2}$, where δ represent the thickness of the hyperplane. The set S_{in} is generated by randomly selecting a subset of k points from the previous digital hyperplane. S_{up} (resp. S_{down}) is generated the same way, by using the same digital hyperplane translated along the (resp. the opposite of the) main direction of n (by more than δ to prevent the convex hulls of the sets to overlap, or less than δ to give a chance for these convex hulls to overlap – in this last case the convex hulls might not overlap due to the fact that the points are picked out randomly, but we do not keep the sets if it happens). An example of two randomly generated sets in 2D is shown in Fig. 4.

⁴ If we denote K_i the vertices of K , a $k-1$ -simplex that is associated to a λ_i coordinate is a $k-1$ -simplex of K that does not contain the vertex K_i .

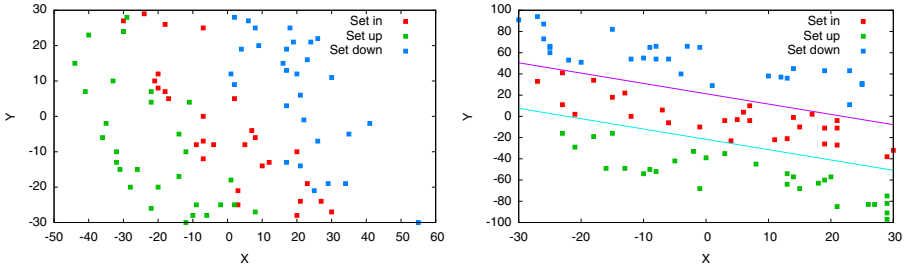


Fig. 4. An example of two typical inputs used to do the experiments. On the left, three random subsets of three parallel digital lines that overlap (no separable affine strip exists). On the right, three other random subsets that are separable by an affine strip (the supporting lines of a solution are drawn in the figure).

Fig. 5 shows the execution times (in ms) of the algorithm according to both separable and non separable sets of points in 4D. All the experiments were carried out on a normal laptop (Core2 Duo 2.20GHz, 4Go RAM). Note that the number of points shown in the figure refers to the size of only one set. But each of the three input sets contains k points, so the algorithm actually deals with $3 \times k$ points. For each size, twenty random inputs have been generated. The vertical lines represent the min, max and the average execution times. We can notice that for both cases (separable and overlapped) the algorithm has a linear run-time behavior (even if it seems faster to detect the non separability of the input sets).

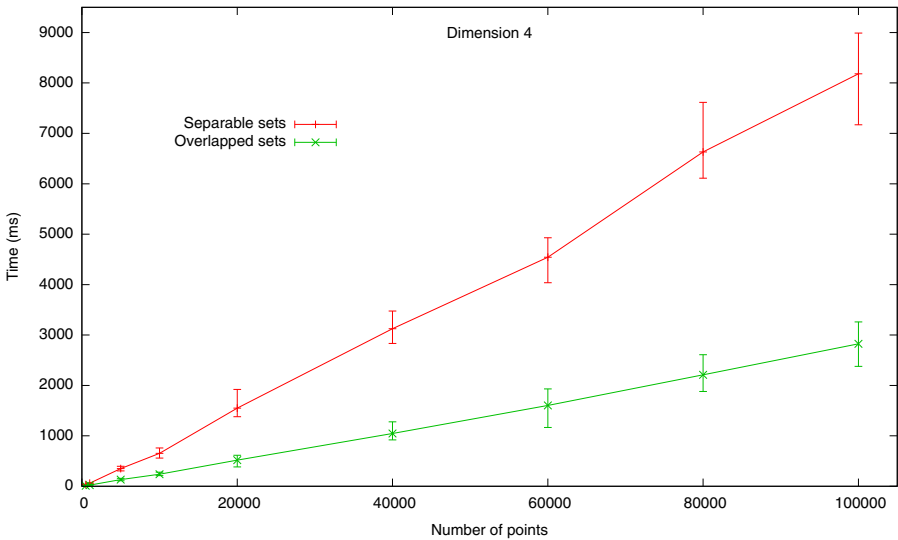


Fig. 5. The execution times of the GJK'nD algorithm on different point sets in 4D

To see the influence of the dimension on the algorithm, similar experiments have been carried out, from dimension 2 up to dimension 10, and the results are given in Fig. 6. A log-scale is used to have a better view of the results. We can notice that the slopes of the curves are quite similar, which means that an increase of the dimension only affects the execution times by a constant multiplicative factor, but the run-time behavior is still linear.

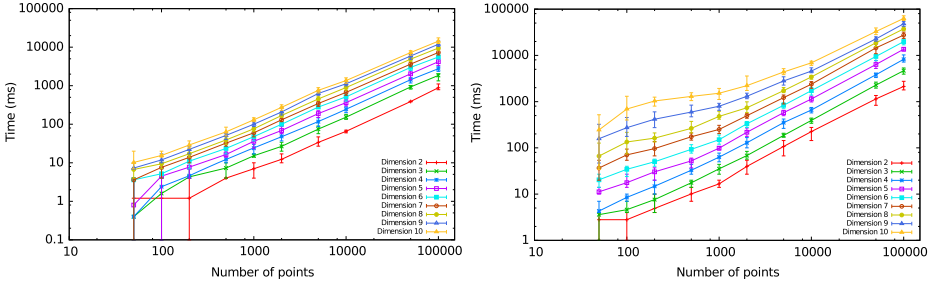


Fig. 6. The execution times (log-scale) of the GJK’*n*D algorithm on different point sets from 2D to 10D. For non separable sets on the left, and separable sets on the right.

4.2 Potential Applications

As stated in the Section 1, an interesting application field to our approach is the recognition of non linear primitives. We show hereafter an example of such applications, namely the recognition of digital annulus, an axis-aligned ellipse and the recognition of an algebraic planar DLL.

Given three point sets S_{in} , S_{up} and S_{down} in \mathbb{Z}^2 the problem is here to find whether there exists a digital annulus that contains S_{in} which encloses the set S_{down} and let S_{up} out of the annulus. It means we are looking for a, b, R^2 and $\delta \in \mathbb{R}$ such that $-\frac{\delta}{2} \leq (x - a)^2 + (y - b)^2 - R^2 \leq \frac{\delta}{2}$ for all $(x, y) \in S_{in}$, $(x - a)^2 + (y - b)^2 - R^2 < -\frac{\delta}{2}$ for all $(x, y) \in S_{down}$ and $\frac{\delta}{2} < (x - a)^2 + (y - b)^2 - R^2$ for all $(x, y) \in S_{up}$. Even if the circle equation in the plane is not linear, we can use the transformation described in Section 2 with $F : (x, y) \mapsto (x, y, x^2 + y^2)$ to get a linear problem in 3D (we actually projected the points onto a paraboloid). We can now use GJK’*n*D to solve this problem in 3D and extract the characteristics of the annulus from the coefficients of the recognized affine strip. Fig. 7 (top-left) shows the result of this method.

This method can be extended to the case of digital axis-aligned ellipses by considering the transformation $F : (x, y) \mapsto (x, y, x^2, y^2)$ which lead to a 4D problem. Furthermore, if S_{in} is given as an 8-connected curve, it is possible to automatically set S_{up} and S_{down} as the inner and outer 4-neighbors of points of S_{in} , as illustrated in Fig. 7 (right). In the same vein, it is worth mentioning that our GJK variant is still well-suited to recognize basic naive digital hyperplanes. In this case we can indeed automatically choose S_{up} and S_{down} wrt. to S_{in} . In 3D, for instance, we just have to take $S_{up} = S_{in} + (0, 0, 1)$ (the set S_{down} can even be empty) to recognize naive digital plane whose main axis is Z .

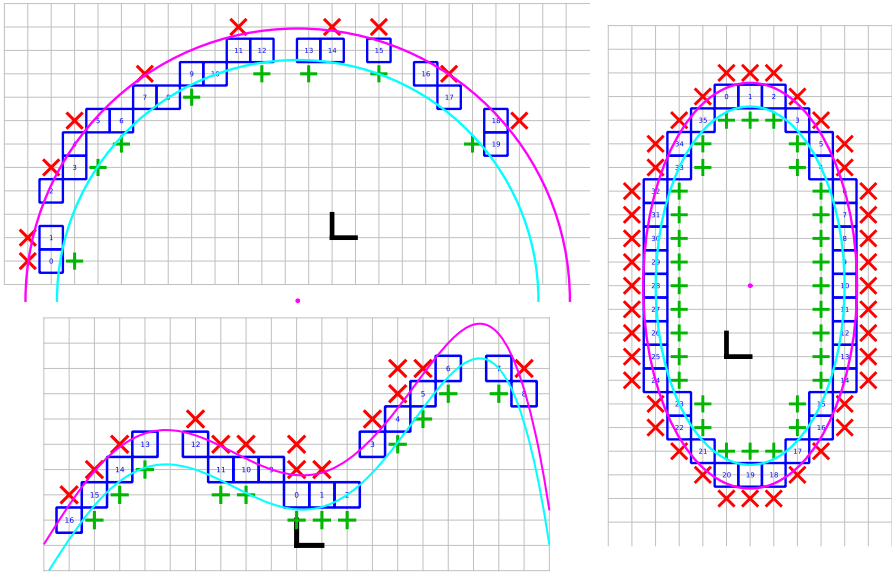


Fig. 7. Top-left, a digital annulus recognition with forbidden points by using the GJK’**n**D algorithm. S_{in} are represented with blue pixels and S_{up} and S_{down} are respectively represented with red and green crosses. Bottom-left, the recognition of an algebraic DLL based on a degree 5 polynomial, and on the right a digital axis-aligned ellipse.

If we are now interested in the recognition of a DLL associated to an underlying polynomial model, i.e. $-\delta \leq y - P(x) < \delta$ with, say, $P(x) = \sum_{i=0}^5 a_i x^i$, we just need to consider another transformation, namely $F : (x, y) \mapsto (x, y, x^2, x^3, x^4, x^5)$. We then have a problem in 6D, but the flexibility of GJK’**n**D allows us to solve this problem. A result is shown is Fig 7(bottom-left).

5 Conclusion

This work comes from the idea that linear digital primitives have been intensively investigated in the last years and that it is time to try to increase the degree of the primitives (not only with circles). DLL are a possibility but one of their drawbacks is that their recognition with a fixed maximal thickness is not suitable for applications because it can introduce undesired neighbors. Hence, we have suggested a more flexible approach where the set to recognize is given along with two sets of outliers instead of a maximal algebraic thickness which has no more geometrical meaning for non-linear primitives.

This new problem is a generalization of the classical problems of recognition of digital hyperplanes: the classical version of the problem corresponds to the case where S_{up} is the translation of S_{in} by a vertical vector of length the maximal authorized thickness and S_{down} is empty. The problem generalizes also the

recognition of digital disks, and thus of their border, namely digital circles. It means that this problem with two sets of outliers allows to handle several classical problems. This approach is very flexible and the solution that we provide by using GJK is rather efficient in practice. Hence, it is a catch-all algorithm that can be improved in some specific sub-problems but which can recognize, with a unique code, a lot of classical or original digital primitives.

References

1. Brimkov, V.E., Coeurjolly, D., Klette, R.: Digital planarity – A review. *Discrete Applied Mathematics* 155(4), 468–495 (2007)
2. Brimkov, V.E., Dantchev, S.S.: Digital hyperplane recognition in arbitrary fixed dimension within an algebraic computation model. *Image and Vision Computing* 25(10), 1631–1643 (2007)
3. Buzer, L.: An incremental linear time algorithm for digital line and plane recognition using a linear incremental feasibility problem. In: Braquelaire, A., Lachaud, J.-O., Vialard, A. (eds.) *DGCI 2002*. LNCS, vol. 2301, pp. 372–381. Springer, Heidelberg (2002)
4. Charrier, E., Buzer, L.: An efficient and quasi linear worst-case time algorithm for digital plane recognition. In: Coeurjolly, D., Sivignon, I., Tougne, L., Dupont, F. (eds.) *DGCI 2008*. LNCS, vol. 4992, pp. 346–357. Springer, Heidelberg (2008)
5. Coeurjolly, D., Gerard, Y., Reveillès, J.P., Tougne, L.: An elementary algorithm for digital arc segmentation. *Discrete Applied Mathematics* 139(1-3), 31–50 (2004)
6. Debled-Renesson, I., Reveillès, J.P.: A linear algorithm for segmentation of digital curves. *International Journal of Pattern Recognition and Artificial Intelligence* 9(4), 635–662 (1995)
7. Gerard, Y., Debled-Renesson, I., Zimmermann, P.: An elementary digital plane recognition algorithm. *Discrete Applied Mathematics* 151, 169–183 (2005)
8. Gérard, Y., Provot, L., Feschet, F.: Introduction to digital level layers. In: Domengoud, E. (ed.) *DGCI 2011*. LNCS, vol. 6607, pp. 83–94. Springer, Heidelberg (2011)
9. Gilbert, E.G., Johnson, D.W., Keerthi, S.S.: A fast procedure for computing the distance between complex objects in three-dimensional space. *IEEE Journal of Robotics and Automation* 4, 193–203 (1988)
10. Megiddo, N.: Linear programming in linear time when the dimension is fixed. *Journal of the ACM* 31(1), 114–127 (1984)
11. Provot, L., Debled-Renesson, I.: 3D noisy discrete objects: Segmentation and application to smoothing. *Pattern Recognition* 42(8), 1626–1636 (2009)
12. Sivignon, I., Coeurjolly, D.: From digital plane segmentation to polyhedral representation. In: Asano, T., Klette, R., Ronse, C. (eds.) *Geometry, Morphology, and Computational Imaging*. LNCS, vol. 2616, pp. 356–367. Springer, Heidelberg (2003)
13. Zrour, R., Kenmochi, Y., Talbot, H., Buzer, L., Haman, Y., Shimizu, I., Sugimoto, A.: Optimal consensus set for digital line and plane fitting. *International Journal of Imaging Systems and Technology* (2011)

A Simple and Flexible Mesh Parameterization Method

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

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

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

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

Abstract. We describe a method to compute conformal parameterizations with a natural boundary based on a simple differentiable expression measuring angles between edges by using complex numbers. The method can be adapted to preserve metric properties or map textures with constrained positions. Some illustrations are shown to assess the efficiency of the algorithms.

Keywords: parameterization, conformal, constrained texture mapping, natural boundary, minimization.

Introduction

Parameterizations of discrete surfaces are one to one maps from a triangulated surface in 3D to the plane. They are widely used in computer graphics because they allow one to simplify difficult 3D problems in easy 2D tasks. For instance, texture mapping, a very classical such application, boils down to the trivial task of mapping an image on a rectangular domain. They also allow to consider a mesh as the graph of a function from the plane to the 3D space. Such a representation is useful for applications such as morphing, surface fitting, etc.

Many methods have been introduced to compute conformal parameterizations. These parameterizations, by definition, should preserve angles and thus the local aspect of the mesh. Among them, barycentric methods [2,7] fix the boundary of the parameterization on a convex shape and then solve a linear system to find the parameters for the others vertices. These algorithms are very fast but introduce a big length distortion near the boundary. Other methods have then been introduced to find parameterizations with a more natural boundary. However, attempts such as the so called *intrinsic parameterization* [1] or *least square conformal map (LSCM)* method [4] use boundary conditions that are not very natural, leading sometimes to strange unsymmetrical results. The *angle based flattening (ABF)* method [8,9,10] gives better results in this respect. But this method is not easy to adapt when using other constraints such as lengths, areas, positions of specific vertices, etc.

In this paper, we introduce a differentiable expression to compute both the angle and the length ratio between two edges. On the one hand, we use it to define a conformal energy measuring angle distortion of a parameterization in a similar way to ABF method. And as this energy is expressed in terms of the position of the vertices of the parameterization and not the angles, it can be adapted to also preserve metric properties or positional constraints. On the other hand, we use it to define a boundary energy to find parameterizations with natural boundary.

The rest of the paper is organized as follows. In Section 1, we define the angular formula and the resulting conformal energy. In Section 2, we detail the minimization algorithm. Section 3 introduces boundary conditions to guarantee convergence of the algorithm and to determine natural boundary. Examples of area preservation and parameterization with constraints are given in the last section.

1 Minimizing Angle Distortion

1.1 Main Problem

Our goal is to find a parameterization of a triangular or quadrangular mesh, that is to say a flattened version of the mesh. A good parameterization should be as close as possible to the initial mesh, and in particular its faces should look like those of the mesh.

By definition, conformal maps are the applications preserving angles and in consequence they also preserve lengths ratios locally. Therefore, for faces which are small with respect to the whole mesh, faces in a conformal parameterization have almost the same shape as those of the initial mesh. Therefore we look at a *good* flattened mesh as a discretization of a conformal map.

Although continuous conformal maps exist, in general, we cannot find a parameterization whose (linear) faces have exactly the same angles as those of the three dimensional mesh. So a natural approach is to look for the parameterization minimizing the angles distortion. It is used by ABF method which computes the parameterization minimizing the energy

$$\sum (\alpha_i - \beta_i)^2,$$

where the sum is over all the angles α_i of the mesh and the β_i are the corresponding angles in the parameterization. The unknowns of ABF method are not the vertices of the parameterization but the angles β_i . Consequently the method has three main drawbacks.

- They have to reconstruct the parameterization from the angles which is not straightforward.
- Not all the angles configurations can be obtained. For example, for triangles faces the sum of the angles must be π . Thus, it is a constrained minimization problem which is solved with Lagrange multipliers technique.
- Since the variables are the angles, we cannot add other energy terms, for example an energy measuring the area distortion.

1.2 A Differentiable Measure of Angle Distortion

In the sequel we will denote by z_i the vertices of the mesh and by z'_i the corresponding two dimensional parameters. We will also consider the z'_i as complex numbers.

Given a triangular face (z_i, z_j, z_k) , we want a formula giving the angle in z_i that we can easily differentiate. Obviously we cannot use a formula such as

$$\arccos\left(\frac{(z_k - z_i) \cdot (z_k - z_i)}{\|z_j - z_i\| \|z_j - z_i\|}\right)$$

for this reason.

However, if we imagine the three vertices as complex numbers in the plane they define, the coefficient

$$\rho = \frac{z_k - z_i}{z_j - z_i} \in \mathbb{C}$$

measures both the angle and the length ratio between the edges $[z_i, z_j]$ and $[z_i, z_k]$. Thus, the nonnegative number $|z'_k - z'_i - \rho(z'_j - z'_i)|^2$ is a measure of the conformal distortion for the face (z_i, z_j, z_k) from the vertex z_i . Moreover, still using complex numbers, we have nice formulas for the derivatives. Indeed, if $z_i = x_i + i y_i$,

$$\begin{aligned} \frac{\partial}{\partial x'_i} &= 2 \operatorname{Re}\left((-1 + \bar{\rho})(z'_k - z'_i - \rho(z'_j - z'_i))\right), \\ \frac{\partial}{\partial y'_i} &= 2 \operatorname{Im}\left((-1 + \bar{\rho})(z'_k - z'_i - \rho(z'_j - z'_i))\right). \end{aligned}$$

For every face, we only need to define a coefficient ρ on one vertex, to preserve all the angles of the face. Finally we want to minimize the energy

$$H = \sum |z'_k - z'_i - \rho_f(z'_j - z'_i)|^2$$

where the sum is over all the faces $f = (z_i, z_j, z_k)$ of the mesh.

For quadrangular meshes, we could convert each face to two triangles and thus construct a triangular mesh but it increases the number of faces. We rather define the ρ coefficients as the ratios of the diagonals. More precisely, for a face (z_i, z_j, z_k, z_l) , we define

$$\rho = \frac{z_l - z_j}{z_k - z_i},$$

and we minimize the function

$$H = \sum |z'_l - z'_j - \rho_f(z'_k - z'_i)|^2.$$

This definition has the advantage of being more symmetric than the previous one, we do not have to choose the vertex in which we compute ρ . Besides, it leads to discrete conformal maps in the sense of [5], where many theorems and in particular convergence theorems are given.

2 Minimization Algorithm

We cannot obtain satisfying results by minimizing the energy term H as such. Nevertheless, we believe it is important to develop the minimization algorithm first. In particular it allows to understand why we introduce new cost functions in Section 3 and why we choose these terms.

2.1 The Algorithm

We achieve the minimization using a BFGS algorithm, a very efficient quasi-Newton method described in [6]. In particular, it computes an approximation of the second derivatives of the function from the exact gradient. So we only need to compute the first derivatives.

Remark 1. The BFGS method assumes a quadratic behavior around the critical point of the function and thus cannot be used to find the critical point of the ABF Lagrangian.

In practice we start from an initial parameterization whose boundary points are on the unit circle and whose interior points are in $(0, 0)$. And we would like the algorithm to unfold the interior points. An example is shown in Figure 1.

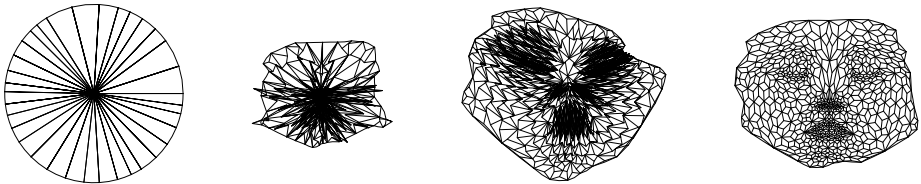


Fig. 1. Four steps of the minimization algorithm, example of good convergence

2.2 Necessity of Additional Constraints

The global minimum of H is 0 and is reached when the z'_i are solutions of the complex linear system :

for all faces $f = (z_i, z_j, z_k)$

$$z'_k - z'_i - \rho_f(z'_j - z'_i) = 0.$$

The number of equations of this system is the number of faces of the mesh whereas its number of unknowns is the number of vertices of the mesh. As a mesh has always more vertices than faces, the system has an infinity of solutions. But many of them are not valid parameterizations.

In fact, if we minimize H as such, the boundary points, initially on the unit circle, pull the interior points to the outside whereas the interior points attract the boundary points to origin. As there are much more interior points than boundary ones, if we do not add any energy, the minimization of H leads to the null parameterization.

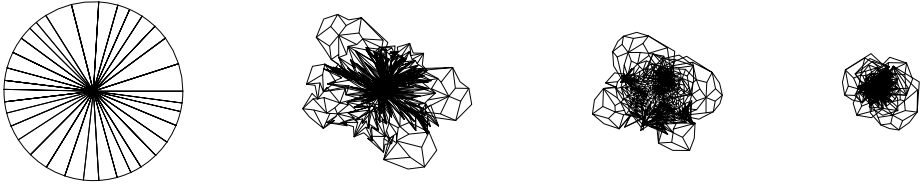


Fig. 2. Four steps of the minimization algorithm, example of bad convergence

3 Stabilizing the Boundary

3.1 Main Idea

The main idea of this section is to add some energy terms to prevent the boundary from being too close to the origin. Constraints should not introduce new distortions. A drastic solution is to fix the boundary points uniformly on a circle. It leads to a very fast and stable minimization process. As all parameterizations methods fixing the boundary, the resulting texture mapping is, in general, unnatural around the boundary. It is particularly true when the boundary of the mesh is very different from fixed boundary of the parameterization.

3.2 Preserving Metric Boundary

Our first motivation is to add an energy to preserve edges lengths around the boundary. Therefore we introduce the following metric energy

$$L = \sum (|z'_i - z'_j|^2 - \|z_i - z_j\|^2)$$

and we propose to minimize an average E of the metric energy and the conformal one

$$E = \alpha H + \beta L,$$

where the coefficients α and β are positive real numbers.

It stabilizes the process for meshes with few vertices and it allows convergence toward a good parameterization. When we increase the number of points, the algorithm can still reach an undesirable local minimum. Thus, the boundary points can be attracted by the origin and wind around it in order to respect boundary edges lengths. An exemple is shown on Figure 2. In this case, using a high value of β compared to α is no more better.

3.3 Preserving Boundary Angles

To prevent the winding around behavior of the algorithm, we propose to add an energy term to preserve angles between border edges. And similarly to conformal energy, in order to obtain a differentiable energy, we introduce complex coefficients measuring both angles and lengths ratios.

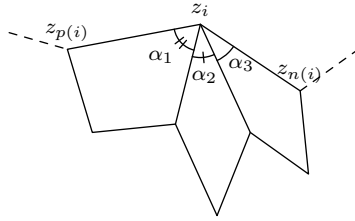


Fig. 3. An example of definition of $\rho = re^{i\theta}$ where $\theta = \alpha_1 + \alpha_2 + \alpha_3$

More precisely, for each boundary vertex z_i , we denote by $p(i)$ and $n(i)$ the indices of the previous and next vertex along the boundary. We want to associate to the vertex z_i a complex number ρ representing the angle between the edges $[z_{p(i)}, z_i]$ and $[z_i, z_{n(i)}]$ and their ratio of lengths. We define ρ as $re^{i\theta}$ where the modulus is

$$r = \frac{\|z_{n(i)} - z_i\|}{\|z_{p(i)} - z_i\|}$$

and the argument θ is the sum of the angles in z_i . In case of Figure 3,

$$\theta = \alpha_1 + \alpha_2 + \alpha_3.$$

Then we introduce the energy

$$B = \sum |z'_{n(i)} - z'_i - \rho_i(z'_i - z'_{p(i)})|^2$$

and minimize

$$E = \alpha H + \beta L + \gamma B$$

for convenient positive real number α , β and γ .

In fact, the energy B is a little redundant with the conformal one. The conformal energy intends to preserve all the angles and in particular the boundary ones. However it is important to strengthen the condition on boundary angles to keep the boundary points from being quickly attracted by the interior points. Thus, at each step of the minimization the interior points remains inside the area delimited by the boundary points allowing convergence toward the right local minimum. Shown in Figure 1 are four intermediate iterations of the algorithm to illustrate the successful convergence.

Remark 2. We could believe it is more natural to define directly θ as the angle between the two edges. But it boils down to choosing an orientation of the plane $(z_{p(i)}, z_i, z_{n(i)})$. Although the mesh is oriented it seems difficult to make a consistent choice.

The resulting algorithm is quite stable in practice. We made many experiments with different meshes and various boundaries, and for most of them, we verify convergence toward the desired parameterization. Moreover the texture mapping are always good near the boundary even with very distorted boundaries.

An example is shown in Figure 4: on the left we represent the parameterization, and on the right, we map a checkerboard on the mesh.

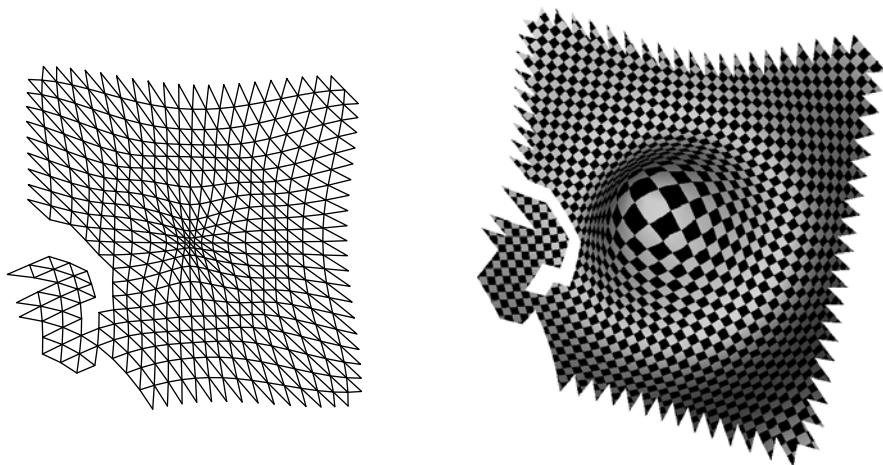


Fig. 4. Parameterization of a triangular mesh with $E = H + B + L$

3.4 Finding a Natural Boundary

The energies B and L of the previous section can also be used to find a natural boundary. Indeed we propose the following three steps algorithm.

1. First we minimize the energy

$$L + B$$

to obtain a boundary with almost the same edges lengths and angles between the edges as the initial boundary of the mesh.

2. Then we fix this boundary and minimize H (the boundary does not move during the algorithm).
3. Finally we use this minimum as initial condition to minimize

$$\alpha H + \beta L + \gamma B.$$

Although this algorithm needs three minimization steps, it is in general faster than the one of the previous section. Step 1 is very fast because the function to minimize has few variables and step 2 too because the boundary is fixed. Finally step 3 is fast because the algorithm start close to the minimum.

An example is shown on Figure 5. Although, the boundary is fixed during step 2, we can see that the texture mapping is quite good. Besides, if we need a fast algorithm to parameterize a mesh with a large number of points and a simple boundary it can be sufficient. The result is better after step 3. The boundary is more natural and the parameterization more conformal. The difference is visible near the chin of the mask To make it clear, Figure 6 shows a zoom in on interesting regions of step 2 and step 3 of Figure 5.

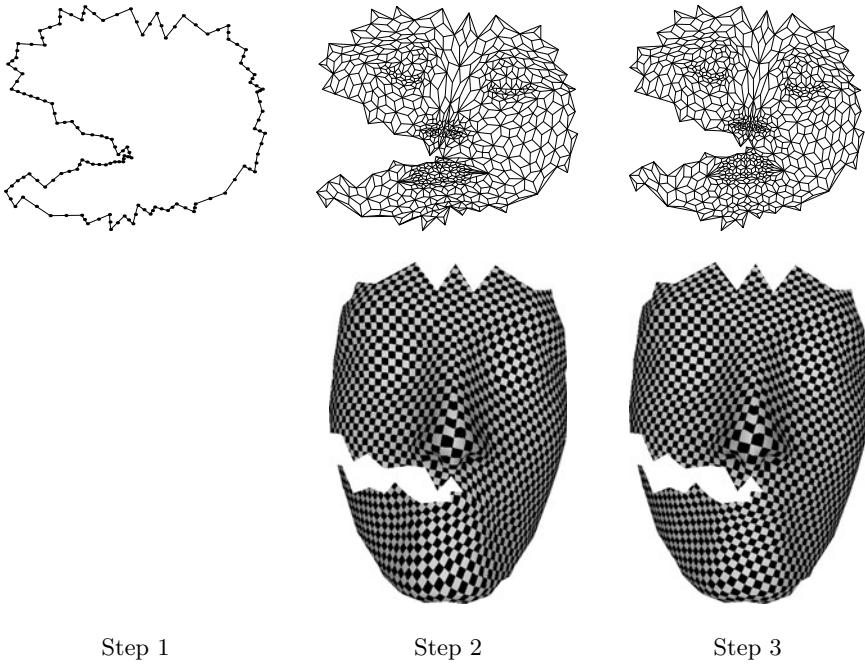


Fig. 5. Parameterizations and texture mappings after each step of the method

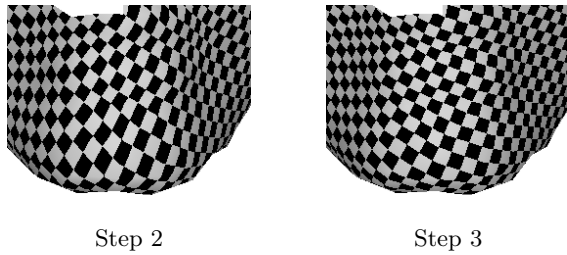


Fig. 6. Zooms in on texture mappings at steps 2 and 3 respectively in Figure 5

4 Introduction of New Constraints

We now further study the advantage of minimizing a conformal energy that expresses in term of points instead of angles. It allows one to add quite easily additional constraints according to the applications.

4.1 Adding Energies: Areas, Lengths Preservation

Even if the main feature of parameterization techniques is to preserve angles, we could also want to preserve other geometrical properties. In particular, on

Figure 4, we would like to reduce the strong metric distortion in regions with high curvature.

In general it is not possible to preserve both angles and lengths. An alternate solution is to relax conformal constraints by adding a new metric energy. Thus, we only obtain a quasi-conformal parameterization but it leads to a better texture mapping.

In case of triangular meshes, to preserve areas we could introduce the energy

$$A = \sum \left(\text{Im}(z'_k - z'_i) \overline{(z'_j - z'_i)}^2 - \|(z_k - z_i) \wedge (z_j - z_i)\|^2 \right)^2$$

where the sum is over all the faces (z_i, z_j, z_k) .

Then, we minimize an average of the form

$$E = \alpha H + \beta A + \gamma L + \delta B$$

The texture mapping of Figure 7 is obtained using such an energy.

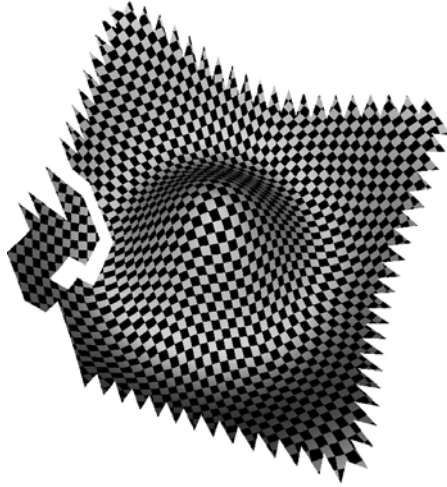


Fig. 7. Parameterization of a triangular mesh with $E = H + A + B$

4.2 Constrained Texture Mapping

Another important application of parameterization techniques is texture mapping of a 2D image on a 3D model. The main features of the image and the model must fit. Therefore positions of the corresponding points of the parameterization must be fixed. Our method can be adapted to that case. In fact it only reduces the number of variables of the function to minimize: we consider the points that are not fixed.

In the example of Figure 9 we map an image of a face on a mask of Nefertiti. We select manually 13 corresponding points. They are displayed with thin points on the figure. On Figure 9 (a), we map the image on the mesh to see that the 13

points are mapped to the right position. On Figure 9 (b) we display the map of a checkerboard with the same parameterization, it shows that the parameterization is still conformal.

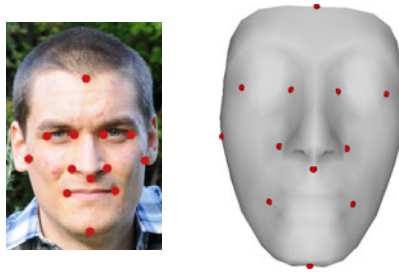


Fig. 8. Corresponding 13 points in the image and the mesh

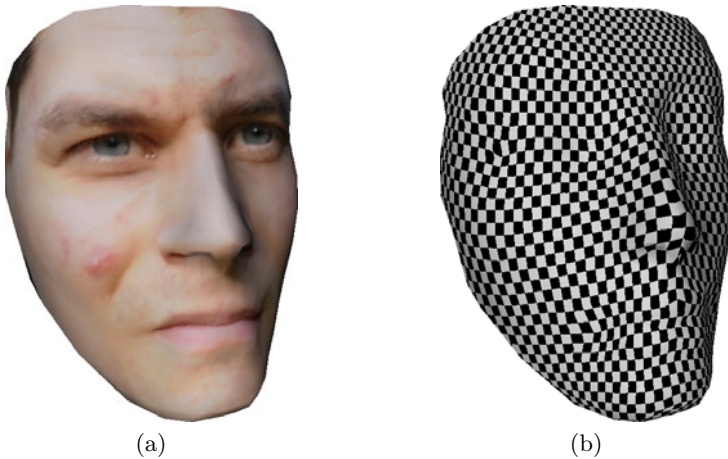


Fig. 9. Textured 3D model with (a) an image face and (b) a checkerboard. Same parameterization computed with $E = H + B$.

5 Conclusion

We have described a method of conformal parameterization of triangular and quadrangular meshes. It consists in introducing a complex number to represent angles between edges and minimizing an energy whose variables are the vertices of the parameterization. Energies measuring angular and metric distortions along the boundary are added to compute parameterizations with natural boundaries. The method is general and can be adapted to preserve areas or to do constrained texture mapping. Many examples are shown to assess the efficiency of our method.

In future work we plan to do a more important comparative study with the state of the art. It involves comparisons of qualitative results and computation times.

Acknowledgments

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

References

1. Desbrun, M., Meyer, M., Alliez, P.: Intrinsic parameterizations of surface meshes. *Computer Graphics Forum* 21, 209–218 (2002)
2. Floater, M.: Mean value coordinates. *Computer Aided Geometric Design* 20(1), 19–27 (2003)
3. Kharevych, L., Springborn, B., Schröder, P.: Discrete conformal mappings via circle patterns. *ACM Transactions on Graphics (TOG)* 25(2), 438 (2006)
4. Lévy, B., Petitjean, S., Ray, N., Maillot, J.: Least squares conformal maps for automatic texture atlas generation. *ACM Transactions on Graphics* 21(3), 362–371 (2002)
5. Mercat, C.: Discrete Riemann surfaces and the Ising model. *Communications in Mathematical Physics* 218(1), 177–216 (2001)
6. Nocedal, J., Wright, S.: Numerical optimization. Springer, Heidelberg (1999)
7. Pinkall, U., Polthier, K.: Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics* 2(1), 15–36 (1993)
8. Sheffer, A., De Sturler, E.: Surface parameterization for meshing by triangulation flattening. In: *Proc. 9th International Meshing Roundtable*. Citeseer (2000)
9. Sheffer, A., De Sturler, E.: Parameterization of faceted surfaces for meshing using angle-based flattening. *Engineering with Computers* 17(3), 326–337 (2001)
10. Sheffer, A., Lévy, B., Mogilnitsky, M., Bogomyakov, A.: ABF++: Fast and robust angle based flattening. *ACM Transactions on Graphics* 24(2), 311–330 (2005)
11. Wegert, E.: Nonlinear Riemann-Hilbert problems—history and perspectives. In: *Computational Methods and Function Theory 1997 (Nicosia)*. Ser. Approx. Decompos, vol. 11, pp. 583–615. World Sci. Publ., River Edge (1999)

Ellipse Constraints for Improved Wide-Baseline Feature Matching and Reconstruction

Dominik Rueß¹ and Ralf Reulke²

¹ Deutsches Zentrum für Luft- und Raumfahrt e.V. in der Helmholtz-Gemeinschaft, Institut für Robotik und Mechatronik, Optische Informationssysteme, Rutherfordstraße 2, 12489 Berlin, Germany

<http://www.dlr.de/os/>

² Humboldt-Universität zu Berlin, Institut für Informatik, Computer Vision, Rudower Chaussee 25, 12489 Berlin, Germany

<http://www.informatik.hu-berlin.de/cv/>

Abstract. The classic feature matching process has two drawbacks. Firstly, ambiguous but possibly correct matches will potentially be removed and secondly, there is no constraint for the 2D size of the features.

In the present paper these drawbacks are tackled at once with a different approach: by considering region features instead of point features and by adding constraints based on the features' shape. Here, the shape will be described with an ellipse. Using existing knowledge about the algebraic properties of ellipses within the computer vision domain, this enables additional constraints such as ellipse tangents. The number of ambiguous matches is reduced and increased control of the physical 2D size of the features is obtained. This will be shown on known epipolar geometry.

Additionally, reconstruction of feature ellipses is examined.

Keywords: Key Points, Feature Regions, Ellipses, Feature Matching, Epipolar Constraints, Reconstruction.

1 Introduction

A major drawback of 2D surveillance systems is inaccuracy when fusing the data of several cameras after being processed. Some reasons are projection onto a plane and occlusions. An example of such a setup is given by Reulke et al [17]. Several cameras in a sensor network are used to track multiple traffic objects to determine atypical events. Data fusion is performed after having extracted object trajectories in image space.

To improve the quality of such results it is possible to perform the surveillance tasks in reconstructed 3D space. It eliminates the tedious task to fuse several 2D trajectories to an erroneous 3D trajectory, for instance. But for this to happen successfully, robust feature matching algorithms have to be developed:

In many multi camera applications there is some sort of feature matching involved. Usually this is performed by detecting features or key points followed

by a computation of some numerical description, e.g. SIFT [8] or MSER [11]. Obtained in both views, these features are matched based on their Euclidean distances in feature space.

Usually the ambiguity at this stage is very high. In many cases, there are several key points with an approximately equal lowest feature distance for a given key point of the other image. The epipolar constraint is a handy tool to improve this situation. For this, the *fundamental matrix* $F \in \mathbb{R}^{3 \times 3}$ of rank 2 is introduced. F maps a homogeneous point in one image to its corresponding homogeneous epipolar line in the second image. This leads to the constraint: Two homogeneous feature locations \mathbf{x} in image 1 and \mathbf{x}' in image 2 correspond to a single point in space, if and only if \mathbf{x} is on the epipolar line $\mathbf{l} = F^T \mathbf{x}'$ or alternatively \mathbf{x}' is on the epipolar line $\mathbf{l}' = F \mathbf{x}$. For more details refer to Ma et al. [9].

This method helps improving the quality of feature matching significantly. Unfortunately, in wide-baseline situations the matching may become worse again. In such a situation the distance of two cameras is relatively high compared to the scene distance. Due to the projective nature of the problem, i.e. occlusions, different backgrounds, different lightning and other problems, corresponding features can look less alike as compared to small baseline situations and again more and more ambiguities may arise.

At this point, the present paper will incorporate additional constraints. First of all, the features will be assigned a geometric shape. The ellipse will show quite suitable for this as it has a rather simple algebraic description. There are also many useful properties, in the generalized form of a conic section. This allows the introduction of additional epipolar constraints, namely the tangency constraints. The paper will show how these properties can be assembled to add more robustness to feature matching in wide-baseline situations.

The topic of this paper is aimed at the retrieval of better surveillance data, that is “real” 3D data. The paper will have a look at a traditional matching algorithm and provide ways to improve it, such that the number of wrong correspondences decreases.

1.1 Previous Work

Rotation and scale invariant features have become very popular since the introduction of “SIFT” [8]. Ever since, many feature detectors and descriptors have been added. “Maximally Stable Extremal Regions” [11] (MSER) are of particular interest for us, as they provide a robust way of detecting the same regions in different views separated by a wide-baseline. For a comprehensive overview refer to [12] and [13].

One of the first reconstruction of conics in two views to conics in space has been done by Quan [16]. This paper also provides an algebraic matching constraint, where the section of the two corresponding cones decomposes into two planes. This test is fine for noise-free perfectly matched ellipses but unfortunately, in our tests it couldn’t be applied to ellipses which are only close to

match perfectly (as it happens in the case of wide-baseline vision), due to the algebraic nature of the whole background.

An important work on this topic was given by Cross and Zisserman [3] (for more details refer to the PhD thesis of G. Cross [2]). They present methods for reconstruction of quadrics in space, based on conics on the vision sensors. Important for our paper is the tangency conservation under the fundamental matrix F . That is, a tangent from an ellipse in the first image to the epipole will remain a tangent in the second image, for the second epipole – if both ellipses match. Hence, the reconstruction does not consider planar features, which we assume.

More work on the topic of reconstruction algebraic curves has been provided by Kaminski and Shashua, see [6] and [7]. They derive very generic methods for reconstructing planar and non-planar algebraic curves of any order. Some of their theory could be used for the current paper.

One of the newer papers is provided by Mai et al [10], where they reconstruct projective ellipses with minimization techniques. This is not suitable for the present paper, as it works with circles and ellipses directly and not with features.

Two state of the art approaches show how to determine the epipolar geometry, which is supposed to be known in this report. A first method is based on one-dimensional features, meaning points on a line [21]. The second way is to use weighted least squares to actually compute an unbiased estimation of the fundamental matrix, without outliers [22].

2 Background

The basic mathematical background necessary for the purpose of this paper is presented in this section. This includes some information about conics, ellipses, quadrics and transformations. First of all, an ellipse can be described in various ways. For most methods presented in this paper, it will be provided in the algebraic projective form.

The classic algebraic ellipse can be written in a conic equation (see [20]):

$$Ax^2 + 2Bxy + Cy^2 + 2(Dx + Fy) + G = 0 \quad (1)$$

with the conditions

$$\Delta = \begin{vmatrix} A & B & D \\ B & C & F \\ D & F & G \end{vmatrix} \neq 0, \quad J = \begin{vmatrix} A & B \\ B & C \end{vmatrix} > 0, \quad \frac{\Delta}{A+C} < 0 \quad (2)$$

All points \mathbf{x} which satisfy equation (1) are part of this ellipse (or the elliptical conic envelope). Equation (1) can also be written for homogeneous coordinates, with a symmetric matrix:

$$\mathbf{x}^T \cdot E \cdot \mathbf{x} = 0 \quad (3)$$

$$\text{where } E = \begin{pmatrix} A & B & D \\ B & C & F \\ D & F & G \end{pmatrix} \quad (4)$$

Methods to transform ellipses are required, e.g. from image to camera coordinates or to perform a transformation of points on the ellipse. The following lemma will show to be quite useful:

Lemma 1. *Let K be an invertible transformation in $\mathbb{R}P^n$, which transforms a point \mathbf{x} , with $\mathbf{x}^T M \mathbf{x} = 0$, to $\mathbf{x}' = K \mathbf{x}$. Then the quadric M can also be transformed with $M' = K^{-T} M K^{-1}$ such that $\mathbf{x}'^T M' \mathbf{x}' = 0$.*

Proof. Use $\mathbf{x} = K^{-1} \cdot \mathbf{x}'$ and insert into $\mathbf{x}^T \cdot M \cdot \mathbf{x} = 0$

2.1 Parametrization

Later on we will need a parametrization of the ellipse E .

Lemma 2. *The ellipse E as defined in equation 3 can be parametrized with:*

$$t \in [0, 2\pi) : \mathbf{x}(t) = \mathbf{c} + \begin{pmatrix} a \cos(t) \cos(\phi) - b \sin(t) \sin(\phi) \\ a \cos(t) \sin(\phi) + b \sin(t) \cos(\phi) \end{pmatrix} \quad (5)$$

where

$$\phi = \begin{cases} 0 & \text{for } B = 0 \text{ and } A < C \\ \frac{1}{2}\pi & \text{for } B = 0 \text{ and } A > C \\ \frac{1}{2} \cot^{-1}\left(\frac{A-C}{2B}\right) & \text{for } B \neq 0 \text{ and } A < C \\ \frac{\pi}{2} + \frac{1}{2} \cot^{-1}\left(\frac{A-C}{2B}\right) & \text{for } B \neq 0 \text{ and } A > C \end{cases} \quad (6)$$

$$\mathbf{c} = \begin{pmatrix} \frac{CD-BF}{B^2-AC} \\ \frac{AF-BD}{B^2-AC} \end{pmatrix} \quad (7)$$

$$a = \sqrt{\frac{2(AF^2 + CD^2 + GB^2 - 2BDF - ACG)}{(B^2 - AC) \left[\sqrt{(A - C)^2 + 4B^2} - (A + C) \right]}} \quad (8)$$

and

$$b = \sqrt{\frac{2(AF^2 + CD^2 + GB^2 - 2BDF - ACG)}{(B^2 - AC) \left[-\sqrt{(A - C)^2 + 4B^2} - (A + C) \right]}} \quad (9)$$

Proof. Refer to [20].

It does also work the other way around:

Lemma 3. *Let $\mathbf{x}(t)$ be a parametrization as in equation 5. Let further R be defined as:*

$$R = \begin{pmatrix} \cos(\phi) & -\sin(\phi) & c_0 \\ \sin(\phi) & \cos(\phi) & c_1 \\ 0 & 0 & 1 \end{pmatrix} \quad (10)$$

Then the algebraic ellipse E can be computed as

$$E = R^{-\top} \cdot \begin{pmatrix} \frac{1}{a^2} & 0 & 0 \\ 0 & \frac{1}{b^2} & 0 \\ 0 & 0 & -1 \end{pmatrix} \cdot R^{-1} \quad (11)$$

Proof. Start with an axis aligned ellipse, for both the parametrization and projective description. Use any Euclidean movement and lemma [7](#).

The following two lemmas are of major importance for this paper. They show how the tangents of an ellipse can be computed with respect to the epipoles.

Lemma 4. *In the projective space \mathbb{RP}^2 every conic, described by the matrix M , has exactly two tangents which both contain a given point $\mathbf{y} \in \mathbb{RP}^2$. These tangents can be described by the equation*

$$(\mathbf{y}^\top M \mathbf{x})^2 - (\mathbf{x}^\top M \mathbf{x})(\mathbf{y}^\top M \mathbf{y}) = 0 \quad (12)$$

Proof. Refer to [19](#)

Lemma 5. *Let \mathbf{y} be a point in \mathbb{RP}^2 . The tangency points of a conic M with respect to \mathbf{y} can be found in the section of the polar line of \mathbf{y} with respect to M and M itself. The equation of the polar line is given by*

$$\mathbf{y}^\top M \mathbf{x} = 0 \quad (13)$$

Proof. Refer to [19](#).

The last lemma provides the mathematical background for computing tangency points or tangents of an ellipse to an epipole, for instance. To obtain the tangency points, one can compute the polar line, followed by solving a quadratic system of $\mathbf{x}^\top M \mathbf{x}$ including the polar line. As the polar line is linear, the system can be solved by rather easy means.

3 Application to Feature Matching

First of all consider image features being planar. This is only an approximation of the reality. But any feature detector suffers from this problem. Once a non-planar surface is viewed from different viewpoints, the accuracy of the point reconstruction will decrease. And in reality most features are small parts of surfaces, locally approximately planar. Non-planar features will usually drop in the matching process anyways, as soon as the epipolar constraints are violated due to the resulting inaccuracy.

The second important aspect is to assume regions rather than point features. In a wide-baseline setup it is not desirable to search for point features only, because most descriptors are not completely invariant to viewpoint change, scale and rotation. This would not affect vision tasks in a small baseline setup but as the baseline grows the features become more and more ambiguous. To overcome

this, it is advisable to use as much additional information as possible. Here, the geometric shape of the regions, especially the size of the respective fitted ellipse is used. A good overview for ellipse fitting was provided by Gander et al [4].

As a consequence this rules out famous detectors like SIFT and others, which search for point features. A very robust and fast region detector is MSER, which is used here for all experiments (it computes regions in linear time, see [15]).

3.1 Descriptors

For the works of Mikolajczyk and Schmid ([12], [13]) executable binaries can be found in their on-line supplements. These binaries detect feature points and are also able to compute descriptions for these features. We tested the “SIFT” and “GLOH” descriptors (gradient location and orientation histogram, which is similar to SIFT) on MSER regions.

Unfortunately, it can take quite a long time to compute the descriptions. We have experienced up to 10 seconds of computation, based on the number of features and the size of the images. And there are significantly less MSER regions, as it will be demonstrated in the results section. That’s the reason why an own SIFT-like circular descriptor has been implemented. It samples the gradients in a scale space, selecting the level based on the size of the single ellipses. Additionally, dominant angle computation was used, leading to a normalization of the ellipses to circles with a predefined dominant direction. The sampling of gradients into a 3D descriptor is based on the gradients’ properties radial distance, position (angle) and gradient direction. Based on empirical tests and the experience in [13] we chose 3 radial bins, 4 position bins for the two outer circles and 8 gradient direction bins, resulting in a $8 + 2 \times 4 \times 8 = 72$ sized feature vector. These types of descriptions have been described in great detail and can be found in [12] or [13], for instance. The performance of our own descriptors is expected to be worse than the presented ones, sacrificed for speed. But at the same time, it will turn out that the improved matching process compensates this drawback and generally improves the results.

3.2 Matching

The matching is now performed as with classical point features, using basic epipolar geometry conditions, described with the following pseudo code:

1. Determine potential matches,
e.g. using a kd-tree and usual feature distances
2. for all features involved in the potential matching set
determine the ellipses’
 - center
 - and both tangency points with respect to the epipole
3. for all potential matches (e1,e2) ensure three constraints:
 - center2 lies on epipolar line of center1
 - one tangency point of e2 lies on the epipolar line of the

- first tangency point of e_1
- the other tangency point of e_2 lies on epipolar line of the second tangency point of e_1

A proof that tangents of an ellipse and the epipole map to tangents of a corresponding ellipse in a second image can be found in [3] and in [5].

After having discarded many of the potential matches with this method, it is now even possible to perform an assignment problem on the remaining matches, where the overall sum of all matches is maximized.

To solve for the remaining matching problem, there are several possibilities: Firstly, only accept matches where the second best potential match distance is significantly larger (e.g. by a factor λ - this has been done in [8]). Or, secondly, by solving the assignment problem with the remaining potential matches, e.g. [14].

3.3 Non-linear Optimization

Given an elliptic match there are usually inaccuracies in the ellipse conditions, due to image noise and projective effects. That is, the ellipses don't exactly match. Therefore, to reconstruct the respective space ellipses, it is recommended to optimize the ellipse pairs. An optimization is presented in [3], which tries to optimize the ellipse such that both ellipses converge to the tangent epipolar lines of the partner ellipse. Here we present a different approach, where also the centers of both ellipses are considered.

Let

$$f(\mathbf{x}, \mathbf{x}') := d(F\mathbf{x}, \mathbf{x}') + d(F^T \mathbf{x}', \mathbf{x}) \quad (14)$$

be the sum of the distances of two corresponding points to the epipolar line of the respective point, where \mathbf{x} in the first and \mathbf{x}' is in the second image. $d(\mathbf{l}, \mathbf{x})$ is the distance of a point to a line. The following overall expression can be minimized to improve the matching conditions and to obtain exact ellipses E and E' simultaneously:

$$\arg \min_{E, E'} f(\mathbf{c}, \mathbf{c}') + \sum_{i \in \{1, 2\}} f(\mathbf{t}_i, \mathbf{t}'_i) \quad (15)$$

where \mathbf{c} and \mathbf{c}' are the ellipse centers and \mathbf{t}_i and \mathbf{t}'_i are the corresponding tangency point pairs of E, E' .

If equation [15] was minimized directly, using the respective matrices, it would be possible that the shapes change to parabolas or hyperbolas. To restrict the minimization to the space of ellipses, we use equations [5] and [11] for conversion to a parametrized form of the ellipse and vice versa. This procedure is quite similar to the non-linear optimization approaches in [9]. As an example, they have provided a way to optimize a rotation not directly using a matrix $R \in \mathbb{R}^{3 \times 3}$ rather than optimizing a parametrized representation in $SO(3)$ (utilizing Rodrigues' formula). This ensures that any change of the parameters still results in a rotation, as it couldn't be guaranteed by direct manipulation of the matrix entries in R .

When using the parametrized version of the ellipses, a simple gradient descent approach has shown to be sufficient to optimize equation 15.

3.4 Reconstruction

We found two major methods for reconstructing ellipses from two imaging sensors to a planar ellipse in space. The first algebraic approach was presented by Quan 16 and the second approach was provided by Schmid and Zisserman 18, which describes a more epipolar geometry oriented derivation. We implemented the method of Quan, for details refer to the respective paper.

Note that a 2D Ellipse in space can't be described by a quadric. But it can be described by a conic and a transformation into space: Let \mathbf{p} be the vector which describes the plane where the cones of two corresponding ellipses intersect in space, then

$$\mathbf{x}^T E(\mathbf{p}) \mathbf{x} = 0 \tag{16}$$

$$\text{where } E(\mathbf{p}) = \frac{1}{2} \cdot \begin{pmatrix} 0 & 0 & 0 & p_0 \\ 0 & 0 & 0 & p_1 \\ 0 & 0 & 0 & p_2 \\ p_0 & p_1 & p_2 & 2p_3 \end{pmatrix} \tag{17}$$

also describes a homogeneous plane equation. To transform \mathbf{p} into $\mathbf{p}' = (\lambda, 0, 0, 0)^T$ choose a transformation T , such that

$$T^T E(\mathbf{p}) T = E(\mathbf{p}'), \tag{18}$$

$$\text{where } T = \begin{pmatrix} R & \mathbf{t} \\ 0 & 1 \end{pmatrix}. \tag{19}$$

Here, $\mathbf{x}^T E(\mathbf{p}') \mathbf{x} = 0$ equals $2\lambda x_0 = 0$ and thus describes the x -plane in the Cartesian coordinate system.

Given a cone C which connects an ellipse to it's camera center, the ellipse on the section of C with the plane \mathbf{p} in space can be computed by transforming the cone with T :

$$C' = T^T C T \tag{20}$$

and by finally choosing the conic matrix S as upper left 3x3 sub-matrix of C' .

S describes the 2D section ellipse on that section plane, given a transformation T . This section ellipse S and the respective transformation T can be used to visualize the ellipses or to put them into a tracking algorithm. In this case the parametrization can be helpful.

4 Results and Experiments

In this section we will present some experiments utilizing the above theory. In all settings calibrated cameras were used. Interior parameters were determined using

the Brown calibration model [1]. This allows for geometrically very exact image normalization (undistortion). The exterior orientation for all cameras involved was determined by Photogrammetric means, e.g. by utilizing exactly measured points (DGPS) or other methods.

4.1 Outdoor Data Set

The outdoor data set shows two images taken from different viewpoints at the same time. They are taken from a camera system which is able to capture radiometrically very high quality images, every 1 s. Synchronization of the two cameras is provided with a maximum difference of less than 50 ns. The baseline length is approximately 6.7 m. The distance to the scene begins with 20 m and goes as far as about 100 m. The resolution of both cameras is 1360x1024. The scene contains natural areas, e.g. homogeneous parts of grass, as well as man-made structures including repeating patterns. Refer to Figure 1 to see two images of these cameras. The computing unit was equipped with an Intel® Core™2 Duo E8600 processor.

To point out the performance of the presented algorithm, two matching algorithms are compared:

The classical epipolar line matching, where the lowest Euclidean feature distance d_1 wins in case of ambiguities – but only if the feature distance d_2 of the next best pair is of a factor λ larger: $d_1 < \lambda d_2$. And the second method: the three ellipse epipolar lines are considered (tangency points and center), rather than just the center epipolar lines. Again, only matches are acknowledged if $d_1 < \lambda d_2$, in case of ambiguities. As described before, these ambiguities or the number of neighbors in the kd-tree should be significantly less in the second method.

Both algorithms come with pros and cons. The first, classic matching algorithm has been built and used to obtain very reliable matches. So its strength is a very distinct computation of features and reliable retrieval of correspondences. It works with point features. The main drawbacks are the longer feature computation time, the occurrence and rejection of ambiguous matches as well as lack of control of the features' size of the. As you will see, the feature computation on large images takes a very long time. For that reason we took the same feature computation as in the second algorithm. This further reduces the performance of this first algorithm.

The newly introduced, second algorithm emphasizes the constraints on geometric shape; therefore it works with region features. It does not use the most distinctive features, satisfied for faster feature computation times. This is also the drawback of the method: without the help of the geometric constraints it would surely perform worse. But tied together it is supposed to work better than the first version.

Indeed, in table 1 you can see that there are less wrong correspondences of the introduced ellipse epipolar constraints, as compared to classical epipolar line matching. This is confirmed by the number of average and maximum neighbors per feature which satisfy the respective constraints. For the classic matching



Fig. 1. Outdoor Data Set. The input images are a stereo image pair from an outdoor scene. The baseline is ≈ 6.7 m and the distance to the scene somewhere between 20 m to 100 m. The scene contains natural areas and man-made structures (i.e homogeneous and repeating parts).

there are much more ambiguities. Figure 2 shows some of the matched ellipses, which illustrate the principle of the tangent epipolar lines.

Table 1. Result Data for the Outdoor Data Set.

The computation times and other results are shown. Note that the feature computation time of approach [12], [13] is noticeably larger (for way less features). Thus, we did not conduct further experiments.

outdoor data set	classic matching	ellipse matching	features as in [12], [13]
# of features (l/r)	1079/1181	1079/1181	381/372
times (s):			
features	1.18	1.18	6.15
matching	0.068	0.082	-
ellipse optim.	-	0.06	-
# of matches	500	313	-
# of wrong matches	171	25	-
max num neighb.	9	1	-
avg num neighb.	1.92122	0.354958	-

The advantage of the ellipse epipolar matching can also be seen in the wrong correspondences of the classic matching. In Figure 3 you can see a wrong correspondence pair, which is ruled out if ellipse matching was applied.

In Figure 4 you can see a reconstruction of feature ellipses. It turns out that far away ellipses are much larger than close ones. This is due to features in the image; they tend to have the same size but projected into space this results in possibly huge differences.

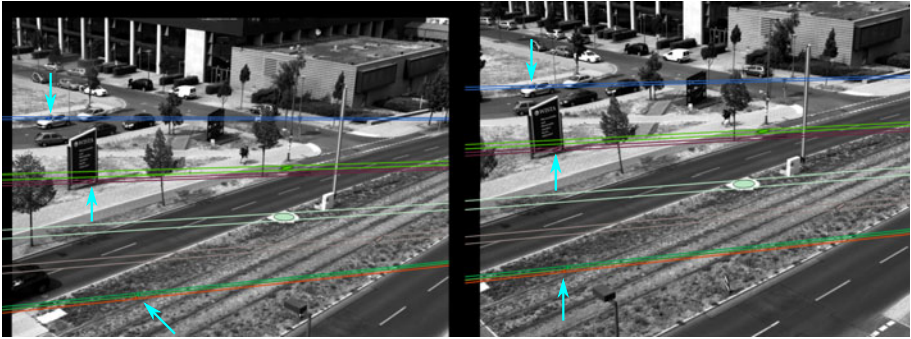


Fig. 2. Outdoor Data Set. Displayed are some of the correspondences after matching using the ellipse epipolar constraints. Smaller features are marked with an arrow. Additionally the tangency epipolar lines of the ellipses are shown.



Fig. 3. Classic Matching: Wrong Correspondences due to Center Constraint only. Two features were matched by classic epipolar line matching (see yellow/dashed epipolar lines). This correspondence is wrong but would have been ruled out by the tangent epipolar lines, as becomes obvious in these images (blue tangent epipolar lines).

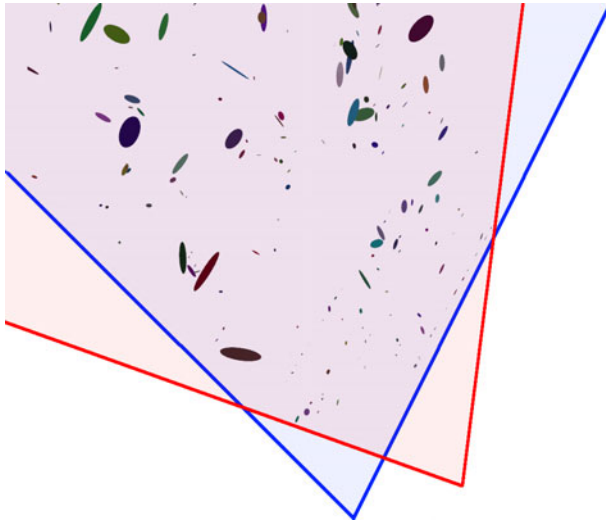


Fig. 4. Ellipse Reconstruction. The reconstructed feature ellipses, viewed from above the scene. Illustrative camera view areas were overlaid. This figure shows how the size of the reconstructed space ellipses is related to the respective distance to the camera positions.

5 Conclusion and Outlook

The present paper has introduced a method to put more epipolar constraints on features by utilizing geometric properties. By requiring that the tangency points of the fitted feature ellipse comply with the epipolar geometry, the number of ambiguities decreases a lot.

It has been shown that this can improve the matching process significantly, especially in an outdoor setup. The loss of time in the matching process is only minor (68 ms vs 82 ms) in comparison to the rest of the process, see table 1.

It has also been shown that “weak” feature descriptors (but potentially faster ones) can be employed. Table 1 shows that the classic epipolar line matching collapses whilst the suggested method still performs very well.

Reconstruction of the ellipse features works but it turns out to be difficult to handle in scenes with high depth variety. The problem is the minimal meaningful area of a region in the image domain. When projected this minimal area becomes huge if the respective depth is high (see figure 4). Also, the ellipse error minimization sometimes seems to reshape the ellipses in a bad way, such that they become elongated in space. Here, more investigation into different error measurements can be done.

A possible application of the presented methods is a 3D feature tracking system for moving traffic objects like pedestrians or cars. We have started working on this and first results look very promising. It currently uses the improved ellipse matching for ambiguity reduction but relies on point reconstruction only.

As described in the last paragraph, reconstructed ellipses tend to become huge farther away from the camera. There is no meaningful relation of the different sizes of the ellipses, which makes it hard to use for tracking. But nevertheless, the center points of the fitted ellipses can be used for accurate reconstruction of the feature regions – with less ambiguities and faster feature computation than before.

Another idea is to investigate into robust calibration methods. In theory, less than 5 conic correspondences (i.e. ellipses) are required to determine the relative orientation (e.g. see [7]). Point based algorithms do need at least 5 correspondences (e.g. see [9]).

Concluding, the matching process can be improved significantly and some sort of control of the size of the features can be gained, if more geometric properties of the feature regions are introduced. This has successfully been done with ellipse feature matching.

References

1. Brown, D.: Close range camera calibration. *Photogrammetric Engineering* 37, 855–866 (1971)
2. Cross, G.: Surface Reconstruction from Image Sequences. PhD thesis, University of Oxford (2000)
3. Cross, G., Zisserman, A.: Quadric reconstruction from dual-space geometry. In: Sixth International Conference on Computer Vision, pp. 25–31 (1998)
4. Gander, W., Golub, G.H., Strebler, R.: Least-squares fitting of circles and ellipses. *BIT Numerical Mathematics* 34, 558–578 (1994)
5. Hartley, R.I., Zisserman, A.: *Multiple View Geometry in Computer Vision*, 2nd edn. Cambridge University Press, Cambridge (2004)
6. Kaminski, J., Shashua, A.: On calibration and reconstruction from planar curves. In: Vernon, D. (ed.) *ECCV 2000*. LNCS, vol. 1842, pp. 678–694. Springer, Heidelberg (2000)
7. Kaminski, J., Shashua, A.: Multiple view geometry of general algebraic curves. *IJCV* 56, 195–219 (2004)
8. Lowe, D.G.: Object recognition from local scale-invariant features. In: *IEEE International Conference on Computer Vision*, vol. 2, p. 1150. IEEE Computer Society, Los Alamitos (1999)
9. Ma, Y., Soatto, S., Košecká, J., Sastry, S.S.: *An Invitation to 3-D Vision*. Springer, Heidelberg (2004)
10. Mai, F., Hung, Y., Chesi, G.: Projective reconstruction of ellipses from multiple images. *Pattern Recognition* 43, 545–556 (2010)
11. Matas, J., Chum, O., Urban, M., Pajdla, T.: Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing* 22, 761–767 (2002); *British Machine Vision Computing*
12. Mikolajczyk, K., Schmid, C.: Scale & affine invariant interest point detectors. *International Journal of Computer Vision* 60, 63–86 (2004)
13. Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27, 1615–1630 (2005)
14. Munkres, J.: Algorithms for the Assignment and Transportation Problems. *Journal of the Society for Industrial and Applied Mathematics* 5, 32–38 (1957)

15. Nistér, D., Stewénius, H.: Linear time maximally stable extremal regions. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part II. LNCS, vol. 5303, pp. 183–196. Springer, Heidelberg (2008)
16. Quan, L.: Conic reconstruction and correspondence from two views. PAMI 18, 151–160 (1996)
17. Reulke, R., Meysel, F., Bauer, S.: Situation analysis and atypical event detection with multiple cameras and multi-object tracking. In: Sommer, G., Klette, R. (eds.) RobVis 2008. LNCS, vol. 4931, pp. 234–247. Springer, Heidelberg (2008)
18. Schmid, C., Zisserman, A.: The geometry and matching of lines and curves over multiple views. International Journal of Computer Vision 40, 199–233 (2000)
19. Semple, J., Kneebone, G.: Algebraic Projective Geometry. Oxford Classic Texts (1998)
20. Weisstein, E.W.: Ellipse. From MathWorld—A Wolfram Web Resource, <http://mathworld.wolfram.com/Ellipse.html>
21. Zhang, Z.: Camera calibration with one-dimensional objects. IEEE Transactions on Pattern Analysis and Machine Intelligence 26, 892–899 (2004)
22. Zhou, H., Green, P.R., Wallace, A.M.: Estimation of epipolar geometry by linear mixed-effect modelling. Neurocomputing 72, 3881–3890 (2009)

Reconstruction of Concurrent Lines from Leaning Points

Peter Veelaert and Michaël Heyvaert

University College Ghent, Engineering Sciences - Ghent University,
Schoonmeersstraat 52, 9000 Ghent, Belgium
{peter.veelaert,michael.heyvaert}@hogent.be

Abstract. In this paper we formulate an alternative approach to the sketch recognition problem. The figure to be recognized from a sketch is specified as a set of geometric line relationship rules. This approach normally has a high computational cost as it essentially is a non-linear optimization problem. We show that for some cases this cost can be avoided by approaching the geometric relationships testing as a ruler construction. In this paper we formulate the recognition problem and consider the construction of line concurrency relationships in detail. We show that the ruler construction is always possible for non-cyclic concurrency relations.

Keywords: Sketch recognition, digital geometry, geometric concurrency.

1 Introduction

The introduction of smart boards has triggered renewed interest in the recognition of objects from loosely drawn sketches [1,7,5,6]. Sketch recognition is used for the automatic conversion of mechanical and geometrical drawings and diagrams into a beautified digital version [11] or to improve pen based computer interfaces. When the system has to recognize a geometric drawing it has to solve a non-linear optimization problem, a task that, due to its difficulty, is left to a mathematical problem solver [3]. The drawback of this approach is that it restricts the recognition task to simple problems that involve only a few lines, and that the time needed to find a solution is quite unpredictable. One way to avoid non-linear optimization is to limit the recognition to drawings with known icons or symbols that can easily distinguished, i.e. a drawing of an electrical circuit [2]. However, it is difficult to extend these icon-based techniques to geometric drawings, because the domain knowledge of geometry is not iconic in nature, but involves geometric relationships such as collinearity, parallelism and concurrency.

In this paper we show that a certain geometric class of sketch recognition problems can be solved without referring to a mathematical problem solver. A parameter domain determines the position and slope uncertainty of a line. We show that all interesting line configurations can be found by ruler constructions. This construction is always possible, provided the concurrency relations are not cyclic, a concept that will be clarified in the paper. Furthermore, we introduce the notion of line and point construction order. The notion of order is used to predict the complexity of the construction, i.e., the number of iterations the ruler has to be used to find a line configuration that satisfies the concurrency relations.

Section 2 introduces domains, leaning points and lines, and the notion of the order of a point or line. Section 3 formulates the main problem in geometric sketch recognition. In Section 4 we prove the main results of the paper.

2 Domains and Leaning Points

Discrete straight lines can be approached in two distinct ways. In a first view one regards a digital line as a set of points for which there is continuous line that is *passing close* to these points. This is the preferable viewpoint when regarding a digital line as the digitization of a continuous line. Let W be a finite set of points in \mathbb{R}^2 , and let $\tau > 0$ be a positive thickness. Then the domain of lines that pass close to W can be defined as the set of parameters (a, b) that satisfy:

$$-\frac{\tau}{2} \leq y_i - ax_i - b \leq \frac{\tau}{2}, (x_i, y_i) \in W.$$

The thickness τ can be chosen as large as needed to accommodate for the uncertainty of the position of the pixels. For a hand-drawn line one can choose the same thickness as the one that is used to divide a curve into straight line segments.

In a second view one considers a digital line as the straight boundary (or edge) between two image segments [4]. Let $U = \{u_1, \dots, u_n\}$ and $V = \{v_1, \dots, v_m\}$ be two finite and non-empty subsets of \mathbb{Z}^2 such that U can be linearly separated from V . Then the domain of the lines separating these two regions is now defined as

$$y_i - ax_i - b \geq 0, (x_i, y_i) \in U \tag{1}$$

$$y_j - ax_j - b \leq 0, (x_j, y_j) \in V. \tag{2}$$

A domain becomes infinite if some of the separating lines are vertical lines. To avoid this complication, in what follows we will assume that there is at least one slope a_V that does not appear in any of the domains involved in the sketch recognition problem. In that case, we can always rotate the coordinate axes such that the lines with slope a_V become vertical lines. After rotation all domains will be bounded convex polygons in the ab -parameter plane.

Fig. 1(a) shows two disjoint point sets U and V . The straight lines that separate these two sets cover a butterfly shaped region. Fig. 1(b) shows the domain that contains the parameters of the separating lines. We will use extensively the dualism between lines and points in the xy -plane and the ab -plane. The point (a, b) in the ab -parameter plane represents the line $y = ax + b$ in the xy -plane, and the line $b = \beta - \alpha a$ in the parameter plane represents the point (α, β) in the xy -plane. For domains of lines we will make these notions more specific.

Definition 1. Let D be a convex, polygonal domain of line parameters with vertex set V_D , and edge set E_D . Any straight line of the form

$$y = a_i x + b_i, (a_i, b_i) \in V_D,$$

is called a *leaning line* of D . A point (x, y) that satisfies

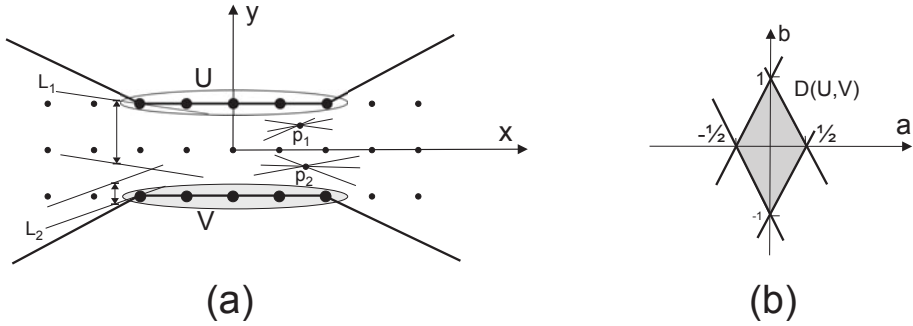


Fig. 1. (a) Straight lines that separate two discrete sets U and V , and (b) the parameter domain of all the separating lines

$$\begin{cases} y = a_i x + b_i \\ y = a_j x + b_j, \end{cases} \tag{3}$$

for an edge $\{(a_i, b_i), (a_j, b_j)\}$ of E_D , is called a *leaning point* of D .

The intersection of two leaning lines is not necessarily a leaning point, unless the leaning lines correspond to adjacent vertices of D . If in (3) we have $a_i = a_j$, then (3) has no solution and there is no leaning point. Likewise, a straight line passing through two leaning points is not necessarily a leaning line, unless the leaning points correspond to edges that share a vertex of D .

The concurrency relations will be solved by ruler constructions that start from leaning points. Given a set of leaning points and leaning lines we construct new points and lines by applying either the meet \wedge or the join operator \vee . Let $L_1 \wedge L_2$ denote the intersection of the straight lines L_1 and L_2 , and let $p_1 \vee p_2$ denote the straight line that joins the points p_1 and p_2 . The number of times an operator has been used to construct a certain point or line determines its order.

Definition 2. Let D be a domain. The leaning lines and leaning points of D have order 1. When new lines and points are constructed by taking joins or meets, an order is assigned as follows:

- If p is not a leaning point, the order of p is the smallest number $n > 1$ for which we can construct at least two lines L_1 and L_2 of order $n - 1$ or less such that $p = L_1 \wedge L_2$.
- If L is not a leaning line, the order of L is the smallest number $n > 1$ for which we can construct at least two points p_1 and p_2 of order n or less such that $L = p_1 \vee p_2$.

Note that the order only increases when we construct intersection points, but not when we construct lines through points. For example, a point that is not a leaning point, but an intersection point of two leaning lines, has order 2. However, a line that is not a leaning line, but that passes through two leaning points, still has order 1. Figure 2 shows a construction in the xy -plane. The black dots represent the leaning points of 5 distinct domains. All the leaning points have order 1. The lines \tilde{L}_1 and \tilde{L}_2 are leaning lines and

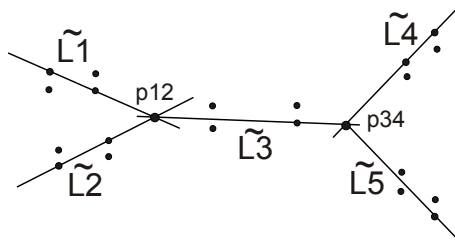


Fig. 2. Illustration of line and points orders

have order 1. The point p_{12} was constructed as the intersection of \tilde{L}_1 and \tilde{L}_2 , and has order 2. The line \tilde{L}_3 was constructed by joining p_{12} to one of the leaning points of the third domain. Therefore, \tilde{L}_3 has order 2. The point p_{34} lies at the intersection of \tilde{L}_3 and a leaning line, and therefore p_{34} has order 3. The line \tilde{L}_5 has order 3 because it was constructed by joining p_{34} to one of the leaning points of the fifth domain.

3 Geometric Figure Recognition

The line $y = ax + b$ will be denoted as $L_{(a,b)}$.

Definition 3. Let $C = \{D_1, \dots, D_n\}$ be a collection of n domains. A line configuration in C is a collection of n lines $L_{(a_i,b_i)}$ such that $(a_i, b_i) \in D_i$.

Definition 4. Let $L_{(a_i,b_i)}, \dots, L_{(a_k,b_k)}$ be m lines chosen from a line configuration. A geometric relation is an equation of the form $P_j(a_i, \dots, b_k) = 0$, where P_j is a real polynomial in the $2m$ variables a_i, \dots, b_k .

Parallelism, perpendicularity and concurrency are the primary examples of geometric relations that can be expressed by a polynomial:

- Two straight lines $L_{(a_1,b_1)}, L_{(a_2,b_2)}$ are **parallel** when $a_1 - a_2 = 0$. We shall denote the involved polynomial as $P_{par}(a_1, b_1, a_2, b_2) = a_1 - a_2$.
- Two lines $L_{(a_1,b_1)}, L_{(a_2,b_2)}$ are **perpendicular** when $a_1 a_2 + 1 = 0$. We shall denote the involved polynomial as $P_{perp}(a_1, b_1, a_2, b_2) = a_1 a_2 + 1$.
- Three lines $L_{(a_1,b_1)}, L_{(a_2,b_2)}$ and $L_{(a_3,b_3)}$ are **concurrent** if

$$\begin{vmatrix} 1 & a_1 & b_1 \\ 1 & a_2 & b_2 \\ 1 & a_3 & b_3 \end{vmatrix} = 0.$$

or $P_{conc} = 0$ with $P_{conc}(a_1, \dots, b_3) = -a_2 b_1 + a_3 b_1 + a_1 b_2 - a_3 b_2 - a_1 b_3 + a_2 b_3$.

Our main problem can now be formulated as follows. Suppose we are given n domains D_i and m geometric relations P_j . Can we find a line configuration $L_{(a_1,b_1)}, \dots, L_{(a_n,b_n)}$, with $(a_i, b_i) \in D_i$ such that $P_j(a_i, \dots, b_k) = 0$ holds for all m relations P_j ?

In its fullest generality this is a non-linear programming problem with linear inequalities (the domains) and non-linear equalities (the geometric relations for perpendicularity and concurrency). However, if the problem is restricted to parallel relations, then it is still relatively easily to solve, since only the slope intervals of the lines count [8,10]. In this paper we will focus on the more difficult case of concurrency.

4 Constructions with Leaning Points and Leaning Lines

Geometrically, finding concurrencies in the xy -plane is equivalent to finding common stabblings in the ab -plane [8,9]. When 3 or more parameter points lie on a stabbing line in the ab -plane, the corresponding lines in the xy -plane are concurrent. If the stabbing line has the form $b = \beta - \alpha a$, the common intersection point is (α, β) .

Figure 3 illustrates this duality between stabbing lines and common intersection points. Figure 3(a) shows 5 distinct pairs of sets U_i and V_i . Figure 3(b) shows the corresponding domains in the ab -plane of lines that separate the sets U_i and V_i . The vertices of the domains correspond to the leaning lines that are shown in Figure 3(a). The edges correspond to the leaning points, indicated by larger dots in Figure 3(a). In this example we want to find lines $L_{(a_1,b_1)}, \dots, L_{(a_5,b_5)}$ that satisfy the concurrency relations $P_{conc}(a_1, b_2, a_2, b_2, a_3, b_3) = 0$ and $P_{conc}(a_3, b_3, a_4, b_4, a_5, b_5) = 0$.

Therefore, in the ab -plane we must find common stabblings for D_1, D_2, D_3 and D_3, D_4, D_5 . What complicates the stabbing problem is that the intersection point of the two stabblings has to lie in D_3 . This is not guaranteed merely by the existence of the two stabblings. Their intersection point may lie outside D_3 . The results that follow will show that it is nonetheless possible to construct lines that satisfy all the requirements. These lines are constructed from the leaning points by a straightedge (or ruler) construction. In addition, we prove that the order of the constructed lines is finite. Since the number of leaning points is also finite, it follows that all possible candidate solutions can be verified in a finite way.

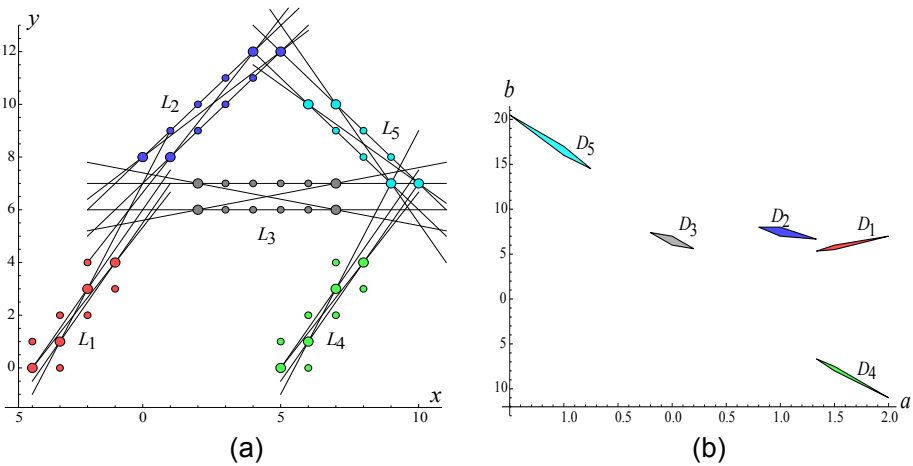


Fig. 3. Stabbling relations in the ab -plane correspond to concurrency relations in the xy -plane

Proposition 1. *Let D_1, D_2, D_3 be three disjoint domains. Suppose there is a line configuration $L_{(a_1, b_1)}, L_{(a_2, b_2)}, L_{(a_3, b_3)}$ with $(a_i, b_i) \in D_i$ satisfying the concurrency relation $P_{conc}(a_1, \dots, b_3) = 0$. Then there also exists a straight line configuration $L_{(\tilde{a}_1, \tilde{b}_1)}, L_{(\tilde{a}_2, \tilde{b}_2)}, L_{(\tilde{a}_3, \tilde{b}_3)}$ with $(\tilde{a}_i, \tilde{b}_i) \in D_i$ and satisfying $P_{conc}(\tilde{a}_1, \dots, \tilde{b}_3) = 0$, such that two of the lines are leaning lines of two distinct domains, and the third line passes through at least one leaning point of the third domain.*

Proof. The proof is based on successive constructions in the ab -plane. Since the lines L_1, L_2, L_3 satisfy the concurrency relation $P_{conc}(a_1, \dots, b_3) = 0$ they meet at a common intersection point (x_0, y_0) . Because each of the lines $y = a_i x + b_i$ pass through (x_0, y_0) , the parameter line $b = y_0 - a x_0$ stabs each of the domains D_i in the ab -plane. We will consider the position of the vertices of the domains with respect to this stabbing line. For each domain D_i with vertices $(c_{ij}, d_{ij}), j = 1, \dots$, let D_i^+ be the set of vertices for which $d_{ij} \geq y_0 - c_{ij} x_0$, and let D_i^- be the set of vertices (c_{ij}, d_{ij}) for which $d_j \leq y_0 - c_j x_0$. First, consider the following linear system in the unknowns α, β :

$$\begin{aligned} d_{ij} &\geq \beta - c_{ij}\alpha, (c_{ij}, d_{ij}) \in D_1^+ \cup D_2^+ \\ d_{ij} &\leq \beta - c_{ij}\alpha, (c_{ij}, d_{ij}) \in D_1^- \cup D_2^- \\ b_3 &= \beta - a_3\alpha. \end{aligned} \tag{4}$$

This system determines the line parameters (α, β) of all the lines in the ab -plane that pass through (a_3, b_3) and that stab both D_1 and D_2 . The solution set is the intersection of a convex polygon and a straight line. Hence, it is a line segment in the $\alpha\beta$ -plane. Let (α', β') be one of the endpoints of this line segment. Then the line $b = \beta' - \alpha\alpha'$ passes through at least one vertex of either D_1 or D_2 . Without loss of generality we assume that it passes through a vertex of D_1 , which we shall denote as $(\tilde{a}_1, \tilde{b}_1)$.

Next we look for a parameter line that also passes through a second vertex of one of the domains. For the vertices $(c_{ij}, d_{ij}) \in D_3$, let D_3^+ be the set of vertices for which $d_j \geq \beta' - c_j \alpha'$, and let D_3^- be the set of vertices for which $d_j \leq \beta' - c_j \alpha'$. We then consider the linear system

$$\begin{aligned} d_{ij} &\geq \beta - c_{ij}\alpha, (c_{ij}, d_{ij}) \in D_2^+ \cup D_3^+ \\ d_{ij} &\leq \beta - c_{ij}\alpha, (c_{ij}, d_{ij}) \in D_2^- \cup D_3^- \\ \tilde{b}_1 &= \beta - \tilde{a}_1\alpha. \end{aligned}$$

This second system determines the parameters of the lines in the ab -plane that pass through a vertex of D_1 and that stab both D_2 and D_3 . Let (α'', β'') be one of the two vertices of this line segment. Then the line $b = \beta'' - \alpha\alpha''$ passes through a vertex of D_1 and at least one vertex of either D_2 or D_3 . Without loss of generality we assume that it passes through a vertex of D_2 , which we shall denote as $(\tilde{a}_2, \tilde{b}_2)$.

Finally, let $(\tilde{a}_3, \tilde{b}_3)$ be a parameter point on $b = \beta'' - \alpha\alpha''$ that lies also on one of the edges of D_3 . All three parameter points $(\tilde{a}_1, \tilde{b}_1), (\tilde{a}_2, \tilde{b}_2)$ and $(\tilde{a}_3, \tilde{b}_3)$ lie on a common line $b = \beta'' - \alpha\alpha''$ in the ab -plane. Therefore the three lines $y = \tilde{a}_i x + \tilde{b}_i$ are concurrent. Furthermore, two of the parameter points are vertices of domains, and thus correspond to two leaning lines. The third parameter point was chosen such that its line passes through at least one leaning point. \square

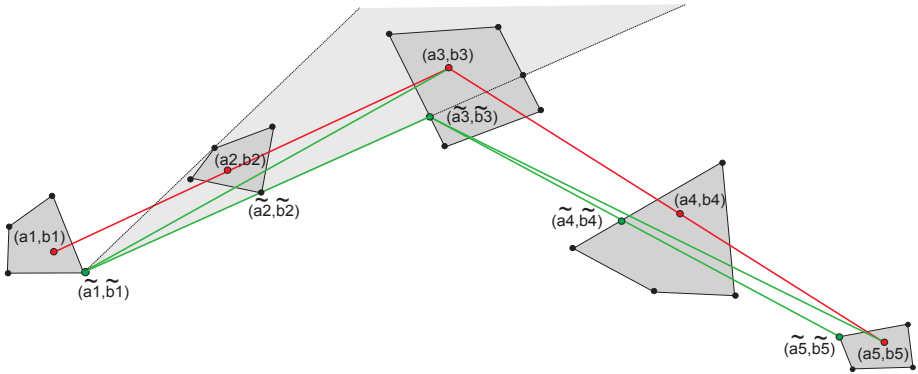


Fig. 4. Illustration of line and points orders

Since leaning lines have order one, their intersection points have order 2. Furthermore, a line passing through such an intersection point and a leaning point also has order 2. Thus the previous proposition states that all lines in the configuration have order 2 or less. This result can be generalized.

Proposition 2. *Let D_1, \dots, D_5 be five disjoint domains. Suppose there is a line configuration $L_{(a_1, b_1)}, \dots, L_{(a_5, b_5)}$ with $(a_i, b_i) \in D_i$ satisfying the concurrency relations $P_{conc}(a_1, b_1, a_2, b_2, a_3, b_3) = 0$ and $P_{conc}(a_3, b_3, a_4, b_4, a_5, b_5) = 0$. Then there also exists a straight line configuration $L_{(\tilde{a}_1, \tilde{b}_1)}, \dots, L_{(\tilde{a}_5, \tilde{b}_5)}$ satisfying $P_{conc}(\tilde{a}_1, \dots, \tilde{b}_3) = 0$ and $P_{conc}(\tilde{a}_3, \dots, \tilde{b}_5) = 0$, with $(\tilde{a}_i, \tilde{b}_i) \in D_i$, such that each line $L_{(\tilde{a}_i, \tilde{b}_i)}$ has order 3 or less with respect to the leaning points of the domains.*

Proof. The proof consists of several steps, and is illustrated in Figure 4.

First concurrency relation. Since the lines satisfy concurrency relations, the points $(a_1, b_1), (a_2, b_2), (a_3, b_3)$ lie on a common straight line in the ab -plane. Also $(a_3, b_3), (a_4, b_4)$ and (a_5, b_5) lie on a common straight line. These two lines intersect at (a_3, b_3) . By constructing the system 4 as in Proposition 1 we determine either $(\tilde{a}_1, \tilde{b}_1)$ or $(\tilde{a}_2, \tilde{b}_2)$. Suppose we have determined $(\tilde{a}_1, \tilde{b}_1)$, then this parameter point yields a parameter line $b = \beta' - \alpha\alpha'$ that passes through the points $(\tilde{a}_1, \tilde{b}_1)$ and (a_3, b_3) , and that stabs the domains D_1, D_2, D_3 , as shown in Figure 4.

Iteration step. Let $H(D_2, (\tilde{a}_1, \tilde{b}_1))$ denote the visual hull of D_2 as seen from the point $(\tilde{a}_1, \tilde{b}_1)$. Since the domains are disjoint convex polygons, $H(D_2, (\tilde{a}_1, \tilde{b}_1)) \cap D_3$ is a convex polygon. Furthermore, this convex polygon contains (a_3, b_3) because the line $b = \beta' - \alpha\alpha'$, which passes through (a_3, b_3) , lies in the visual hull.

Let V denote the vertex set of $H(D_2, (\tilde{a}_1, \tilde{b}_1)) \cap D_3$. Let $\{D_3^+, D_3^-\}$ be a partitioning of the vertex set V with respect to the parameter line passing through the points (a_5, b_5) and (a_3, b_3) . Likewise, let $\{D_4^+, D_4^-\}$ denote the partitioning of the vertex set of D_4 with respect to this parameter line. We consider the linear system

$$\begin{aligned} d_{ij} &\geq \beta - c_{ij}\alpha, (c_{ij}, d_{ij}) \in D_4^{'+} \cup D_3^{'+} \\ d_{ij} &\leq \beta - c_{ij}\alpha, (c_{ij}, d_{ij}) \in D_4^{'-} \cup D_3^{'-} \\ b_5 &= \beta - a_5\alpha. \end{aligned}$$

This system defines a line segment in the $\alpha\beta$ -plane. Let (α'', β'') be one of the two vertices of this line segment. Then the parameter line $b = \beta'' - \alpha\alpha''$ stabs the domains D_3 , D_4 and D_5 . Furthermore, it passes through at least one vertex of either D_4 or $H(D_2, (\tilde{a}_1, \tilde{b}_1)) \cap D_3$. If $b = \beta'' - \alpha\alpha''$ passes through a vertex of $H(D_2, (\tilde{a}_1, \tilde{b}_1)) \cap D_3$, then we will denote this vertex as $(\tilde{a}_3, \tilde{b}_3)$. On the other hand, if $b = \beta'' - \alpha\alpha''$ passes through a vertex of D_4 , then we will denote this vertex as $(\tilde{a}_4, \tilde{b}_4)$. Figure 4 illustrates the case where $b = \beta'' - \alpha\alpha''$ passes through $(\tilde{a}_3, \tilde{b}_3)$ and (a_5, b_5) .

Last concurrency relation. Assume that we have selected $(\tilde{a}_3, \tilde{b}_3)$ in the previous step. Let $\{D_4^{''+}, D_4^{''-}\}$ and $\{D_5^{''+}, D_5^{''-}\}$ be partitionings of the vertex sets of respectively D_4 and D_5 with respect to $b = \beta'' - \alpha\alpha''$. We consider the linear system

$$\begin{aligned} d_{ij} &\geq \beta - c_{ij}\alpha, (c_{ij}, d_{ij}) \in D_4^{''+} \cup D_5^{''+} \\ d_{ij} &\leq \beta - c_{ij}\alpha, (c_{ij}, d_{ij}) \in D_4^{''-} \cup D_5^{''-} \\ \tilde{b}_3 &= \beta - \tilde{a}_3\alpha. \end{aligned}$$

This system yields a new parameter line $b = \beta''' - \alpha\alpha'''$ that stabs the domains D_3 , D_4 and D_5 . Furthermore, this lines passes through $(\tilde{a}_3, \tilde{b}_3)$ as well as a vertex of either D_4 or D_5 . If it passes through a vertex of D_5 , we will denote this vertex as $(\tilde{a}_5, \tilde{b}_5)$. This is the case shown in Figure 4 where $b = \beta''' - \alpha\alpha'''$ is the line joining $(\tilde{a}_5, \tilde{b}_5)$ and $(\tilde{a}_3, \tilde{b}_3)$. If the line passes through a vertex of D_4 , we denote this vertex as $(\tilde{a}_4, \tilde{b}_4)$.

Alternatively, if in the iteration step $(\tilde{a}_4, \tilde{b}_4)$ was selected, then we use the system

$$\begin{aligned} d_{ij} &\geq \beta - c_{ij}\alpha, (c_{ij}, d_{ij}) \in D_3^{''+} \cup D_5^{''+} \\ d_{ij} &\leq \beta - c_{ij}\alpha, (c_{ij}, d_{ij}) \in D_3^{''-} \cup D_5^{''-} \\ \tilde{b}_4 &= \beta - \tilde{a}_4\alpha. \end{aligned}$$

to select a vertex of either D_3 or D_5 . In this case $\{D_3^{''+}, D_3^{''-}\}$ is the partitioning of the vertices of $H(D_2, (\tilde{a}_1, \tilde{b}_1)) \cap D_3$ with respect to the line $b = \beta'' - \alpha\alpha''$.

Trace back step. In the previous steps we have selected three vertices as parameter points $(\tilde{a}_i, \tilde{b}_i)$, one for the first concurrency relation, and two parameter points for the second concurrency relation.

In this final step, we fill in all the other parameters. Suppose, for example, that we have selected the parameter points $(\tilde{a}_1, \tilde{b}_1)$, $(\tilde{a}_3, \tilde{b}_3)$, and $(\tilde{a}_5, \tilde{b}_5)$, as in Figure 4. Then the parameter point $(\tilde{a}_4, \tilde{b}_4)$ can be chosen as an intersection point of the boundary of D_4 and the line passing through $(\tilde{a}_3, \tilde{b}_3)$, and $(\tilde{a}_5, \tilde{b}_5)$. The parameter point $(\tilde{a}_2, \tilde{b}_2)$ can be chosen as an intersection point of the boundary of D_2 and the line passing through $(\tilde{a}_1, \tilde{b}_1)$, and $(\tilde{a}_3, \tilde{b}_3)$. Note that in Figure 4 $(\tilde{a}_2, \tilde{b}_2)$ coincides with a vertex of D_2 , because for $(\tilde{a}_3, \tilde{b}_3)$ we have selected a point that lies on the boundary of the visual hull $H(D_2, (\tilde{a}_1, \tilde{b}_1))$.

As a second example, suppose that we have selected the parameter points $(\tilde{a}_1, \tilde{b}_1)$, $(\tilde{a}_4, \tilde{b}_4)$, and $(\tilde{a}_5, \tilde{b}_5)$. Then the parameter point $(\tilde{a}_3, \tilde{b}_3)$ is chosen as an intersection

point of the boundary of $H(D_2, (\tilde{a}_1, \tilde{b}_1)) \cap D_3$ and the line passing through $(\tilde{a}_4, \tilde{b}_4)$, and $(\tilde{a}_5, \tilde{b}_5)$. Next, $(\tilde{a}_2, \tilde{b}_2)$ is chosen as an intersection point of the boundary of D_2 and the line passing through $(\tilde{a}_1, \tilde{b}_1)$, and $(\tilde{a}_3, \tilde{b}_3)$.

Order of selected points. It remains to determine upper bounds for the order of the selected parameter points. A parameter point can either be vertex of a domain, a vertex of $H(D_2, (\tilde{a}_1, \tilde{b}_1)) \cap D_3$, or an intersection point of the boundary of a domain and a parameter line passing through a vertex of a domain and a vertex of $H(D_2, (\tilde{a}_1, \tilde{b}_1)) \cap D_3$.

Since the boundary edge of a visual hull in the ab -plane passes through two vertices of a domain, it represents an intersection point of order two in the xy -plane, i.e., the intersection point of two leaning lines. It follows that the intersection of a boundary edge of a visual hull and the boundary edge of a domain represents a straight line in the xy -plane of order two. The intersection point of a line of order two and a leaning line yields an intersection point of order 3. \square

Figure 2 shows the same construction order as the construction that was used in Figure 4 to illustrate the above proof. In accordance with Proposition 2 lines \tilde{L}_1 and \tilde{L}_2 have order 1. Line \tilde{L}_3 has order 2, and \tilde{L}_5 has order 3. Proposition 2 states that if a solution exists, it can always be found in this way. Figure 5 gives the solution of the problem that was stated earlier in Figure 3. Figure 5(b) shows the two common stabbings and the selected parameter points. Figure 5(a) shows the lines that correspond to the selected parameters. In this case, all lines are leaning lines and have therefore order 1.

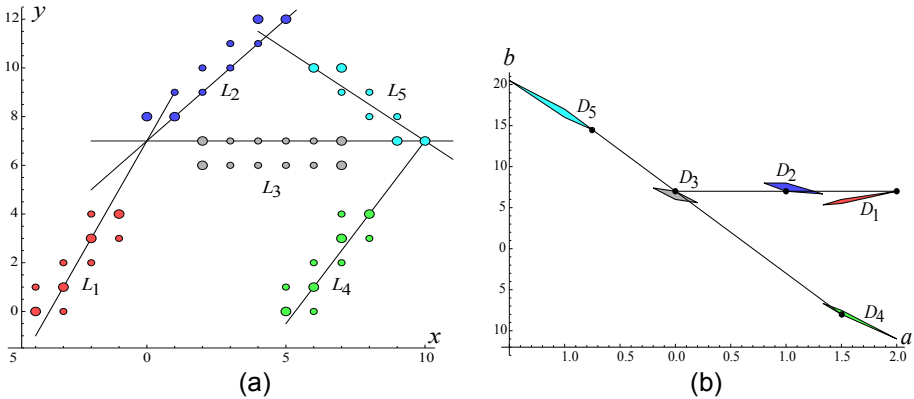


Fig. 5. A solution for the concurrency problem introduced in Figure 3

More generally, suppose a set of domains and concurrency relations is given in the form of Proposition 2. According to Proposition 2, to find a solution it suffices to construct all possible lines of order 3 or less. This line set includes the leaning lines, the second order lines that pass through intersections of leaning lines, and the third order lines that pass through intersections of second and first order lines. In this set it suffices to select those line configurations that satisfy one or more concurrency relations.

Proposition 3. Let D_1, \dots, D_{2N+3} be $2N + 3$ disjoint domains, and let P_1, \dots, P_N be N concurrency relations of the form

$$P_{conc}^k(a_{2k+1}, b_{2k+1}, a_{2k+2}, b_{2k+2}, a_{2k+3}, b_{2k+3}) = 0,$$

for $k = 1, \dots, N$. Suppose there is a line configuration $L_{(a_1, b_1)}, \dots, L_{(a_{2N+3}, b_{2N+3})}$ with $(a_i, b_i) \in D_i$ satisfying the N concurrency relations $P_{conc}^k(a_{2k+1}, \dots) = 0$. Then there also exists a straight line configuration $L_{(\tilde{a}_1, \tilde{b}_1)}, \dots, L_{(\tilde{a}_{2N+3}, \tilde{b}_{2N+3})}$, with $(\tilde{a}_i, \tilde{b}_i) \in D_i$ that satisfies the N concurrency relations $P_{conc}^k(\tilde{a}_{2k+1}, \dots) = 0$ and such that each line $L_{(\tilde{a}_i, \tilde{b}_i)}$ has order $N + 1$ or less with respect to the leaning points of the domains.

Proof. The proof proceeds in the same way as in Proposition 2. With the first concurrency relation we determine either $(\tilde{a}_1, \tilde{b}_1)$ or $(\tilde{a}_2, \tilde{b}_2)$. Next, we apply the iteration step to determine either $(\tilde{a}_3, \tilde{b}_3)$ or $(\tilde{a}_4, \tilde{b}_4)$. Suppose we have determined $(\tilde{a}_3, \tilde{b}_3)$. Then the iteration step is used a second time. That is, we construct the visual hull $H(D_4, \tilde{a}_3, \tilde{b}_3)$ and the vertex set of $H(D_4, \tilde{a}_3, \tilde{b}_3) \cap D_5$. The system

$$\begin{aligned} d_{ij} &\geq \beta - c_{ij}\alpha, & (c_{ij}, d_{ij}) &\in D_5^+ \cup D_6^+ \\ d_{ij} &\leq \beta - c_{ij}\alpha, & (c_{ij}, d_{ij}) &\in D_5^- \cup D_6^- \\ b_7 &= \beta - a_7\alpha. \end{aligned}$$

is used to determine either $(\tilde{a}_5, \tilde{b}_5)$ or $(\tilde{a}_6, \tilde{b}_6)$. Next, the iteration step is repeated with either $H(D_6, (\tilde{a}_5, \tilde{b}_5))$ or $H(D_5, (\tilde{a}_6, \tilde{b}_6))$ and so on. The iteration proceeds until we have determined two parameter points in the N -th concurrency relation. Finally, in the trace back step, all the remaining parameter points are selected at the boundaries of the domain. \square

Figure 6 shows an example where we have determined the parameter points $\tilde{p}_i = (\tilde{a}_i, \tilde{b}_i)$ in the following order: $\tilde{p}_2, \tilde{p}_3, \tilde{p}_5, \tilde{p}_7, \tilde{p}_6, \tilde{p}_4, \tilde{p}_1$.

Proposition 3 does not cover all given sets of concurrency relations. Figure 7 shows an example with three concurrency relations. Suppose we want to find a line configuration in

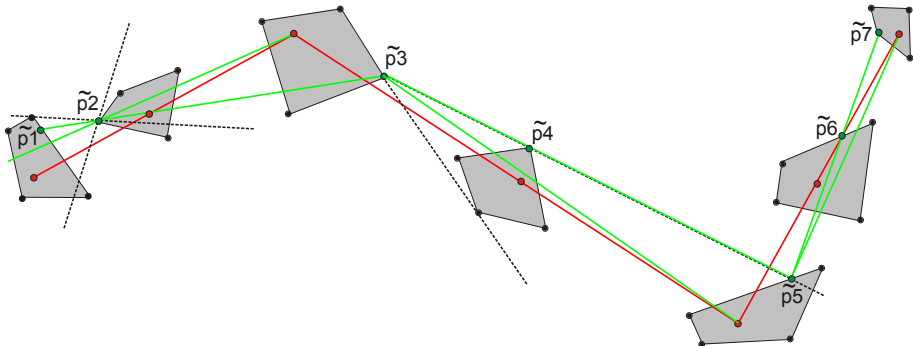


Fig. 6. Illustration of the construction used in Proposition 3 for 3 concurrency relations and 7 domains. The visual hulls constructed from \tilde{p}_2 as well as \tilde{p}_3 are indicated by dashed lines.

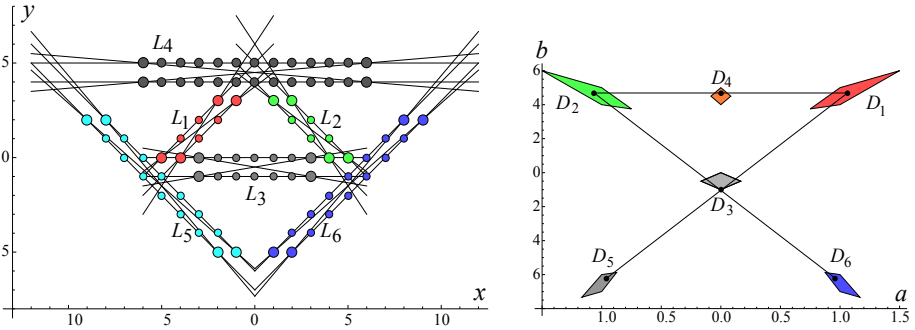


Fig. 7. Three way concurrency with a cyclic dependency

which $\{L_1, L_5, L_3\}$, $\{L_3, L_6, L_2\}$, and $\{L_2, L_4, L_1\}$ are concurrent triples. Figure 7(a) shows the defining sets U_i, V_i , the leaning lines, and the leaning points. Figure 7(b) shows the domains, and three stabbings that would solve the problem. Proposition 3 is not applicable, however. The sequence of concurrency relations ends where it starts, i.e., with the line L_1 . In fact, the proof of Proposition 3 as well as Proposition 2 is explicitly based on the requirement that the sequence of concurrency relations has a loose end where the construction can start.

5 Concluding Remarks

In this work we examine the reconstruction of geometric figures specified by a set of geometric relations. In principle, we could test whether a certain set of detected lines is valid for a line configuration by algorithmically solving a set of non-linear equations and linear inequalities. Even for relatively simple line configurations this approach soon becomes intractable. In this paper we have shown, however, that for a non-cyclic sequence of concurrency relations a solution can always be found in a finite number of steps with a ruler construction, provided such solution exists.

The approach followed differs from previous work on concurrency in digital geometry. In [9] the goal was to derive all the concurrency relations by defining a metadomain as the set of parameters of the lines that stab pairs of domains. Possible candidates for common stabbings of domains were then found by finding cliques in the intersection graph of the metadomains. There was no guarantee, however, that this heuristic approach would always lead to the recognition of a consistent set of geometric relations. Additional constraints were used to remove some of the inconsistencies. In the present paper, the geometric results are consistent by construction. The continuous lines that are constructed must always satisfy a consistent set of relations. The current limitation of the present method is that it has not been proven yet that the method can discover all possible concurrency relations.

In the near future we plan to remove these limitations as much as possible, and also to combine the recognition of concurrent, parallel and perpendicular line relations.

References

1. Don, L., Ivrişimtzis, I.: Multi-pen sketch recognition in a learning environment. *Journal of Multimedia* 4(2), 80–86 (2009)
2. Gennari, L., Kara, I., Stahovich, T., Shimada, K.: Combining geometry and domain knowledge to interpret hand-drawn. *Computer and Graphics* 29, 547–562 (2005)
3. Hammond, T., Davis, R.: LADDER: A sketching language for user interface developers. *Computer and Graphics* 29, 518–532 (2005)
4. Heyvaert, M., Veelaert, P.: Combining geometric edge detectors for feature detection. Accepted for Proceedings of the 11th International Conference on Advanced Concepts for Intelligent Vision Systems (2010)
5. Hse, H., Newton, A.R.: Sketched symbol recognition using Zernike moments. In: Proceedings of the 17th International Conference on Pattern Recognition, pp. 367–370 (2004)
6. Mahoney, J.V., Fromherz, M.: Three main concerns in sketch recognition and an approach to addressing them. In: AAI Spring Symposium on Sketch Understanding, AAI TR SS-02-08 (2002)
7. Ouyang, R., Davis, R.: A visual approach to sketched symbol recognition. In: Proceedings of the 21st International Joint Conference on Artificial Intelligence, pp. 1463–1468. Morgan Kaufmann, San Francisco (2009)
8. Veelaert, P.: Algorithms that measure parallelism and concurrency of lines in digital images. In: Proc. of SPIE's Conference on Vision Geometry VIII, vol. 69, pp. 69–79. SPIE, Denver (1999)
9. 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)
10. Veelaert, P.: Graph-theoretical properties of parallelism in the digital plane. *Discrete Applied Mathematics* 125, 135–160 (2003)
11. Zeleznik, R.C., Herndon, K., Hughes, J.F.: SKETCH: An interface for sketching 3D scenes. In: Proceedings of Siggraph, pp. 163–170. ACM, New York (1996)

Isoperimetrically Optimal Polygons in the Triangular Grid

Benedek Nagy^{1,*} and Krisztina Barczi²

¹ Department of Computer Science, Faculty of Informatics, University of Debrecen,
PO Box 12, 4010 Debrecen, Hungary

² Institute of Mathematics, University of Debrecen, PO Box 12,
4010 Debrecen, Hungary

`nbenedek@inf.unideb.hu`, `bkrixta@googlemail.com`

Abstract. It is well known that a digitized circle doesn't have the smallest (digital arc length) perimeter of all objects having a given area. There are various measures of perimeter and area in digital geometry, and so there can be various definitions of digital circles using the isoperimetric inequality (or its digital form). Usually the square grid is used as digital plane. In this paper we use the triangular grid and search for those (digital) objects that have optimal measures. We show that special hexagons are Pareto optimal, i.e., they fulfill both versions of the isoperimetric inequality: they have maximal area among objects that have the same perimeter; and they have minimal perimeter among objects that have the same area.

Keywords: Discrete isoperimetric problem, digital geometry, digital circles, triangular grid.

1 Introduction

In the Euclidean geometry the isoperimetric inequality shows the privileged role of the Euclidean circles. It states that the area enclosed by a closed simple curve is the largest when enclosed by a circle of the same length, with equality occurring only for circles. This implies two conclusions, as two sides of a coin. In one hand, among closed simple curves of a certain length, a circle encloses a maximal area. On the other hand, among curves enclosing a certain area, a circle has minimal length.

In digital geometry, i.e., in grids (tessellations of the plane) there are some phenomena which do not occur in the Euclidean plane. For instance, there are neighbor points (pixels), etc. There is another important example of non-correspondence of concepts [16]: A digitized circle (see [3,21]) doesn't have the smallest (digital arc length) perimeter of all objects that have a given area.

* The work is supported by the TÁMOP 4.2.1/B-09/1/KONV-2010-0007 project. The project is implemented through the New Hungary Development Plan, co-financed by the European Social Fund and the European Regional Development Fund.

For discrete spaces there are special shapes that have been proved to have minimal ‘perimeter’, for various definitions of the perimeter. In \mathbb{Z}^n , Wang and Wang [25] presented an ordering of grid points, so that every finite prefix of the sequence forms a set with minimal boundary size for that cardinality. Similar arguments have been applied to other classes of spaces by Bezrukov [2]. There are other related results, see [4,9,14]. The results on the hexagonal grid are applied in chemistry also [5,8]. In [10] a correspondence is established between perfect matchings in certain classes of benzenoid graphs and paths in the rectangular lattice that satisfy certain diagonal constraints. A closely related problem in graph theory is the vertex isoperimetric problem: to minimize the number of vertices of the outer boundary. The edge isoperimetric problem, that is to minimize the number of outgoing edges, is completely solved for various types of graphs (see, e.g., [6,12]).

We call a *polygon optimal* if both of the two constraints, to have maximal area among the grid-polygons of a certain length, and to have minimal perimeter among the grid-polygons enclosing a certain area, are fulfilled. Note that in discrete space these two constraints are more or less concurrent, and so, usually do not coincide. In [24] these polygons are called Pareto-optimal, since in game-theory [22] when the aims of the players concur the optimal solution is the saddle point. In that paper results are presented for the square and for the hexagonal grid with long and difficult proofs.

In digital geometry spaces consist of points that can be addressed by integer coordinate values. The square and cubic grids are well-known and frequently used in applications, since they use the Cartesian coordinate system. To use other grids, one needs a good method to define an appropriate coordinate system. The pixels of the hexagonal grid can be addressed with two integers [13]. There is a more elegant solution using three coordinate values whose sum is zero reflecting the symmetry of the grid [11]. Similarly the triangular grid can be described with three values [18,19] as we recall in Figure 1. There are two orientations of the used triangles, the sum of coordinate values are zero and one, respectively.

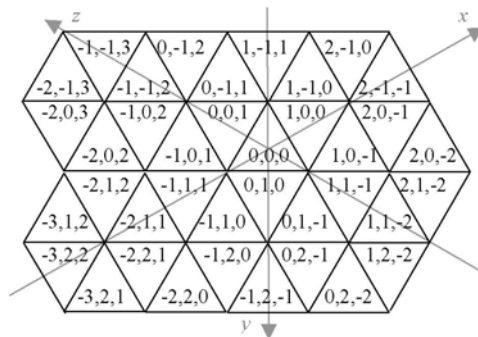


Fig. 1. A part of the triangular grid with a symmetric coordinate frame

In this paper we recall some results of [10,24] (regarding isoperimetric inequality in the square grid), moreover we give an alternative (and simpler) proof of the result based on combinatorics. Our main result is an extension to the triangular grid. Since the triangular grid is more sophisticated, we present our result only by using the closest neighborhood. There is a straightforward possible extension which is to continue this line of research involving larger neighborhood systems.

We may call a finite set of grid points (pixels) a *binary picture* or a *grid-polygon*, or simply an *object*.

We omit the proof of the trivial facts that optimal objects are connected and topologically have no holes (i.e., simply connected). We are dealing only with these objects in this paper. The aim of this paper is to find those objects that have maximal area among those that have at most the same perimeter and, at the same time, they have minimal perimeter among those objects that have at least the same area.

In the next section we recall results on the square grid with new proofs. In Section 3 we present some basic concepts on the triangular grid that are used in Section 4 where our main results are presented. In Section 5 some concluding remarks follow. References and proof of the main theorem (in Appendix) close the paper.

2 Preliminary Results: The Square Grid

There are two types of usual neighborhood relations on the square grid: their original names: city block (or Manhattan) and chessboard neighborhood come from the initial paper on digital geometry by Rosenfeld and Pfaltz [23]. In cellular automata theory the terms Moore and von Neumann neighborhood are used [15]. We also note here that in some cases the terms 1-neighbors and 2-neighbors are used meaning the number of coordinate values that may differ in various types of neighborhoods (this concept allows simple extension to higher dimensions and to other grids, as we will use later on the triangular grid).

The *area* of the object can easily be measured by the *number of grid-squares* it occupies. However, the perimeter depends on the used neighborhood criterion. Let us see which objects are Pareto optimal, first using 4-neighborhood boundary as perimeter.

2.1 Square Grid with 4-Neighborhood

Let us define the perimeter of a binary picture on the square grid with 4-neighborhood. The *perimeter* of an objects is the *number of grid-squares* that belong to the *4-neighborhood* of an *object* point but do not belong to the object.

By the usual Cartesian coordinate frame, the *embedding rectangle* of an object can also be defined: it is a *diamond* object defined by *four stair-type sides*, i.e. diagonal line-segments in the following way. Let $p(p(1), p(2))$ be a/the object point for which the value $p(1) + p(2)$ is minimal/maximal, then the stair-type ‘line’ consisting of points $q(q(1), q(2))$ with $q(1) + q(2) = p(1) + p(2)$ is two of the

sides of this diamond. The other two sides are defined by the minimal/maximal value of $p(1) - p(2)$ for the object points $p(p(1), p(2))$.

As one can see in Fig. 2 the perimeter does not change, while the area is strictly increasing if side parts (a) and (b) are replaced by side part (c). By iteration one can obtain a ‘diamond shape’ object with four ‘stair-type’ sides. Moreover at the ‘corners’ of these sides there are at most two 4-neighbor pixels on the boundary. In this way, in the original shape, since the area is not maximal for the same perimeter, the optimal objects could be only these diamonds. These objects are called *simple shapes* in [24].

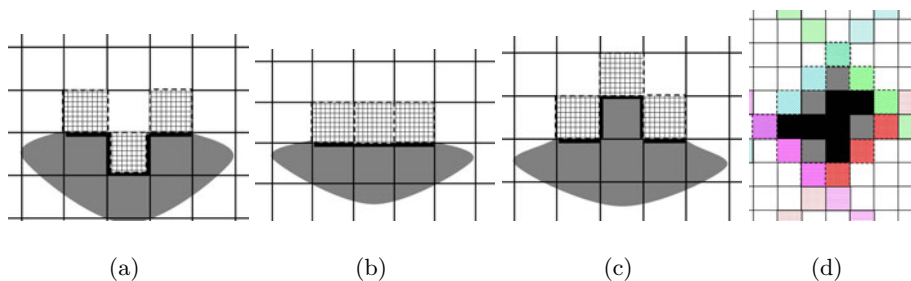


Fig. 2. (a)-(c): Excluding ‘concavity’ and long ‘straight’ sides. (d): An object (black) and its embedding rectangle (grey) with its stair-type sides.

Finally, we need to show what the lengths of the sides of the optimal diamonds are. So let us consider a diamond shape object (see Fig. 2 (d): black and grey pixels). The parameters are the distance of the ‘parallel’ sides and the type of the bottom corner: Let $\Delta x = |\max_x - \min_x|$, where the stair-type sides of direction \setminus are defined by points $r(r(1), r(2))$ with $r(1) + r(2) = \max_x$ and $r(1) - r(2) = \min_x$, respectively. Similarly, let $\Delta y = |\max_y - \min_y|$, where the sides of direction $/$ are defined by points $r(r(1), r(2))$ with $r(1) - r(2) = \max_y$ and $r(1) + r(2) = \min_y$, respectively. Furthermore there are 4 possibilities for the bottom corner (it is at the intersection of lines defined by \min_x and \min_y) by the parity of \min_x and \min_y : *oo* means both are odd, *oe* means \min_x is odd and \min_y is even, similarly *eo* and *ee* is defined.

The perimeter of the diamond is $\Delta x + \Delta y + 4$. (It can be proven by induction on the sidelengths.)

The area of the maximal diamond with these parameters (the area also depends on the bottom corner and so it may be 1 less as one may prove it by combinatorial case analysis): $\lfloor \frac{(\Delta x + 1)(\Delta y + 1) + 1}{2} \rfloor$.

Fixing the perimeter there is only one variable and it gives a maximal value for the area with $\Delta x = \Delta y$.

Based on these values one can easily see that the optimal digital shapes are those where Δx and Δy are (only almost if the perimeter is odd) equal. The results are shown in details in [24].

2.2 Square Grid with 8-Neighborhood

In this case we use a similar argument as in the 4-neighbor case. First we define the *embedding rectangle*: it is a rectangle given by *straight sides* by the lanes of the minimal/maximal $p(1)$ and $q(2)$ values of the points p and q of the object.

From Figure 3(a,b) it is clear that if the object has a ‘corner’ which is concave, then the area can be extended without changing the perimeter.

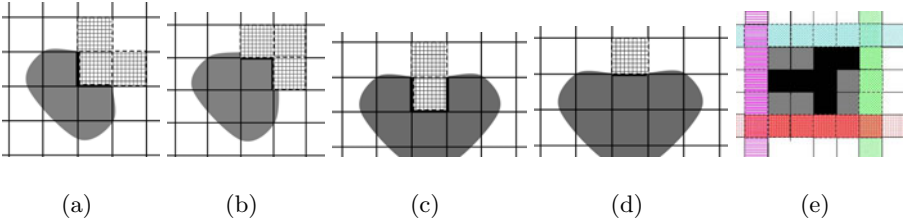


Fig. 3. Excluding concavity (a,b) at corners and (c,d) at sides. (e): An object (black) and its embedding rectangle (grey) with the perimeter points (various color).

Moreover, from Fig. 3(c,d) one can see that a similar argument holds when the concave part is not on the corner, but on one of the sides. The area can be extended without decreasing the perimeter. By combinatory it can easily be proven that there is no other way of concavity to occur.

By iteration, it is obvious that only embedded rectangles (in the usual sense) can be optimal. Also since there must be points at each value of x and y between the maximal and minimal values (see Figure 3 also), actually there are at least two points that can be assigned to each x and y value of the object in its border.

Let us see which of these have the maximal area.

The parameters are $\min_x, \max_x, \min_y, \max_y$, we use the notation $\Delta x = \max_x - \min_x + 1$, and $\Delta y = \max_y - \min_y + 1$.

The perimeter is $2(\Delta x + \Delta y) + 4$.

The area is $\Delta x \Delta y$.

By fixing the perimeter there is only one parameter. By searching for the extremal value (simple derivation): it is at $\Delta x = \Delta y$.

One can easily check that for a perimeter value which is divisible by 4 $\Delta x = \Delta y$ is an integer and so it is optimal in the grid as well.

For perimeter values that are even, but not divisible by 4, in grid the optimal possibilities are those when $\Delta x = \Delta y \pm 1$.

The results of this section can also be found in [24] with a much sophisticated proof. In the next sections, in the triangular grid we prove new results.

3 Definitions and Notions for the Triangular Grid

The triangular grid can be described using a subset of \mathbb{Z}^3 [17][18]. One way of doing it is to take the union of the planes having points with coordinate sums 0

and 1. The points of these two planes are referred as *even/odd* points of the grid, respectively. These two types of points are exactly the type Δ and ∇ triangles of the grid. In this way, the description of the grid is symmetric by the three coordinate values [19,20]. The coordinate axes x, y and z are used.

For instance, an even grid point $(p(1), p(2), -p(1) - p(2))$ has the following closest neighbors $(p(1) + 1, p(2), -p(1) - p(2)), (p(1), p(2) + 1, -p(1) - p(2)), (p(1), p(2), 1 - p(1) - p(2))$. Otherwise, if we consider the points where the sum of the coordinates is 1, the neighbor grid points have difference vectors like the vectors above but with inverted signs.

In this paper only these closest neighbors are used, they are 1-neighbor pixels, since exactly one of the coordinate values changes by stepping from a pixel to one of its 1-neighbors, formally: Let $p = (p(1), p(2), p(3))$ and $q = (q(1), q(2), q(3))$ be two points of the grid. For $k = 1, 2, 3$, the points p and q are *triangular k -neighbors* if $|p(i) - q(i)| \leq 1$ for $1 \leq i \leq 3$ and $\sum_{i=1}^3 |p(i) - q(i)| \leq k$.

Let a coordinate value be fixed (e.g., $x = 5$); a *lane* is the set of points that have this fixed coordinate value.

The area of a binary image on the triangular grid is the number of its triangles. As we use 1-neighborhood boundary, the perimeter of an object is the number of triangles outside of the object that are 1-neighbors of some triangles of the object.

Analogously, as any picture on the square grid has an embedding rectangle, we define the concept of embedding hexagon on the triangular grid:

The embedding hexagon consists of the points that are in the intersection of those lanes (all three directions) which have at least one point of the object (see Fig. 4 for examples). Any picture of the triangular grid has an embedding hexagon. In some cases it can be degenerated with some sides whose length is zero (see Fig. 4 (b,c)).

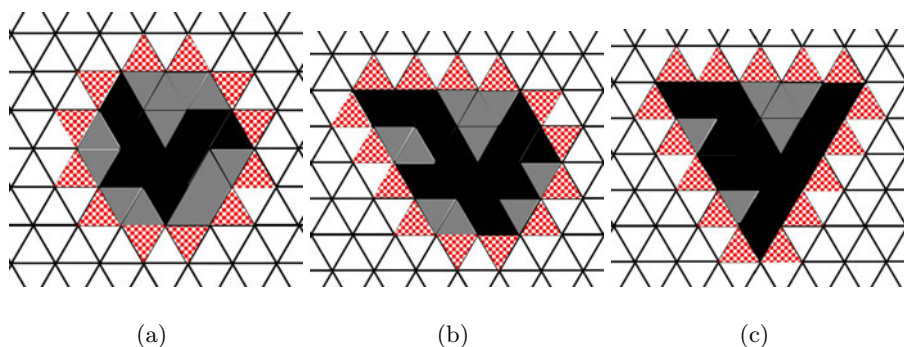


Fig. 4. Embedding hexagons (grey) on the triangular grid (of the black object). The examples (b) and (c) are degenerated. The perimeter of the hexagon is shown by red color.

4 Digital Circles

In this section we present analogous results about Pareto-optimal polygons using the triangular grid. First we will prove that optimal shapes are hexagons, and then we will prove that in optimal polygons the difference of the sidelengths of the hexagon is as small as possible.

4.1 The Shape of Optimal Circles

In this subsection we show that optimal polygons have only straight sides (parallel to sides of the triangles of the grid and there are no ‘hilly’ and ‘sawtooth’ sides in the terms of [20]).

In fact, we will show that the embedding hexagon B (maybe a degenerate version) of a given object A has at most the same perimeter as the perimeter of A , while the area of B is not less than the area of A (it is equal if and only if the objects A and B coincide).

There are two possible types of connection combination of edges that cannot belong to a (degenerated) embedding hexagon; they can be seen on Fig. 5 (a) and (c): concavity can occur only in these two ways having the angle $\frac{5\pi}{6}$ or $\frac{4\pi}{3}$ between edges.

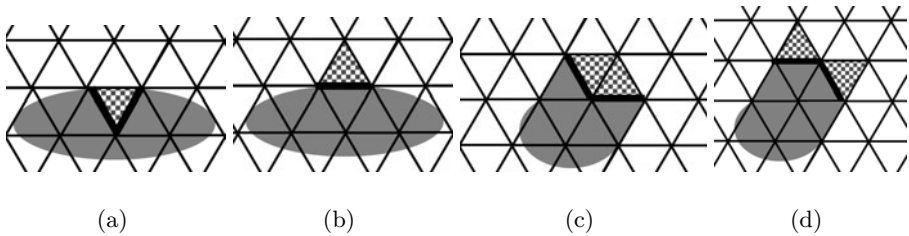


Fig. 5. Excluding concavity in the triangular grid. The object in (a) is blown up to (b) by excluding corner with angle $\frac{5\pi}{6}$, while the object in (c) is extended to (d) by changing an angle $\frac{4\pi}{3}$ to $\frac{2\pi}{3}$ without changing the perimeter.

One can see on Fig. 5 (b) and (d) how these objects can be extended by adding one or two triangles to them, respectively.

By iterative use of the previous local blowing steps the embedding hexagon is obtained. Therefore the optimal shapes can be only the embedding hexagons.

Let the sidelengths of a hexagon be a, b, c, d, e, f in this order, then by the geometry of the grid the lengths of the sides are not independent, but $a + b = d + e$, $b + c = e + f$ and $c + d = f + a$ (note that in degenerate cases one or more values are 0).

In the next subsection we determine the optimal embedding hexagons.

4.2 The Side-Lengths of Optimal Polygons

In continuous case those hexagons are optimal that have equal sides. It cannot be obtained for all possible perimeters in discrete grids. In the triangular grid

there are six cases for the possible perimeters of the hexagons. We present our results by these cases.

Theorem 1. *The next table shows the perimeter and area of the optimal pictures.*

Case	Perimeter	Area
1	$6n$	$6n^2$
2	$6n + 1$	$6n^2 + 2n - 1$
3	$6n + 2$	$6n^2 + 4n$
4	$6n + 3$	$6n^2 + 6n + 1$
5	$6n + 4$	$6n^2 + 8n + 2$
6	$6n - 1$	$6n^2 - 2n - 1$

where n is a natural number such that both the perimeter value and the area are positive.

The proof of the theorem with the optimal objects can be found in the Appendix.

It is not surprising that the optimal hexagons of the continuous case are approximated; increasing the difference of the sides the area is decreasing with a fixed perimeter. One can see that optimal shapes are hexagons with (almost) equal sizes. Fig. 6 shows the perimeter and area of the optimal objects.

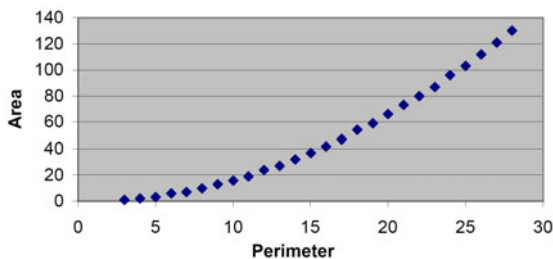


Fig. 6. Pareto optimal values on the triangular grid

5 Conclusions and Future Work

Non-traditional grids are used in image processing and computer graphics. In two dimensions the hexagonal and the triangular grids are the alternatives of the square grid [7,26,27,28]. Pareto optimal objects on the triangular grid using the closest neighbors are presented. It is proven that they are hexagons and the lengths of their sides are as close as possible depending on the perimeter. Our formulae presented in the previous section works for small objects also (see Fig. 6 also). Our optimal shapes are much closer to the Euclidean optimal circles than the rectangles/squares of the square grid. Our problem is closely connected to the vertex-isoperimetric problem of the triangular grid graph and therefore the

results can be applied on that field. In the other side, our optimal polygons can be viewed as results of an optimization process, therefore these results could be applied in some discrete optimization problem.

The result can be extended by using other neighborhood structures, including the other six 2-neighbors and also, the other three 3-neighbors could be included. This is one of our tasks for the near future.

References

1. Altshuler, Y., Yanovsky, V., Vainsencher, D., Wagner, I.A., Bruckstein, A.M.: On Minimal Perimeter Polyminoes. In: Kuba, A., Nyúl, L.G., Palágyi, K. (eds.) DGCI 2006. LNCS, vol. 4245, pp. 17–28. Springer, Heidelberg (2006)
2. Bezrukov, S.L.: Isoperimetric Problems in Discrete Spaces. *Extremal Problems for Finite Sets* 3, 59–91 (1994)
3. Bhowmick, P., Bera, S., Bhattacharya, B.B.: Digital Circularity and Its Applications. In: Wiederhold, P., Barneva, R.P. (eds.) IWCIA 2009. LNCS, vol. 5852, pp. 1–15. Springer, Heidelberg (2009)
4. Bollobás, B., Radcliffe, A.J.: Isoperimetric Inequalities for Faces of the Cube and the Grid. *Europ. J. Combinatorics* 11, 323–333 (1990)
5. Bornhöft, J., Brinkmann, G., Greinus, J.: Pentagon hexagon-patches with short boundaries. *Europ. J. Combinatorics* 24, 517–529 (2003)
6. Brass, P.: On point sets with many unit distances in few directions. *Discrete & Computational Geometry* 19, 355–366 (1998)
7. Brimkov, V.E., Barneva, R.P.: “Honeycomb” vs square and cubic models. *Electronic Notes in Theoretical Computer Science* 46 (2001)
8. Brunvoll, J., Cyvin, B.N., Cyvin, S.J.: More about extremal animals. *Journal of Mathematical Chemistry* 12, 109–119 (1993)
9. Datta, B.: A Discrete Isoperimetric Problem. *Geometriae Dedicata* 64, 55–68 (1997)
10. Doslic, T.: Perfect Matchings in Lattice Animals and Lattice Paths with Constraints. *Croatia Chemica Acta CCACAA* 78(2), 251–259 (2005)
11. Her, I.: Geometric transformations on the hexagonal grid. *IEEE Tr. Image Processing* 4(9), 1213–1222 (1995)
12. Harary, F., Harborth, H.: Extremal Animals. *J. Combin. Inform. System Science* 1, 1–8 (1976)
13. Luczak, E., Rosenfeld, A.: Distance on a hexagonal grid. *Transactions on Computers C-25*(5), 532–533 (1976)
14. Katona, G.O.H.: The Hamming-sphere has minimum boundary. *Studia Scient. Math. Hungarica* 10, 131–140 (1975)
15. Kier, L., Seybold, P., Cheng, C.-K.: *Modeling Chemical Systems Using Cellular Automata*. Springer, Heidelberg (2005)
16. Klette, R., Rosenfeld, A.: *Digital geometry. Geometric methods for digital picture analysis*. Morgan Kaufmann, Elsevier Science, San Francisco, Amsterdam (2004)
17. Nagy, B.: A family of triangular grids in digital geometry. In: 3rd International Symposium on Image and Signal Processing and Analysis, pp. 101–106. IEEE Press, Los Alamitos (2003)
18. Nagy, B.: Generalized triangular grids in digital geometry. *Acta Mathematica Academiae Paedagogicae Nyíregyháziensis* 20, 63–78 (2004)
19. Nagy, B.: A symmetric coordinate frame for hexagonal networks. In: *Theoretical Computer Science - Information Society* 2004, pp. 193–196. ACM, Slovenia (2004)

20. Nagy, B.: Characterization of digital circles in triangular grid. *Pattern Recognition Letters* 25, 1231–1242 (2004)
21. Nagy, B.: An algorithm to find the number of the digitizations of discs with a fixed radius. *Electronic Notes in Discrete Mathematics* 20, 607–622 (2005)
22. Osborne, M.J., Rubinstein, A.: *A course in game theory*. MIT Press, Cambridge (1994)
23. Rosenfeld, A., Pfaltz, J.L.: Distance Functions on Digital Pictures. *Pattern Recognition* 1, 33–61 (1968)
24. Vainsencher, D., Bruckstein, A.M.: On isoperimetrically optimal polyforms. *Theoretical Computer Science* 406, 146–159 (2008)
25. Wang, D.L., Wang, P.: Discrete Isoperimetric Problems. *SIAM Journal of Appl. Math.* 32(4), 860–870 (1977)
26. Wuthrich, C.A., Stucki, P.: An algorithm comparison between square- and hexagonal-based grids. *Graph. Model Im. Proc.* 53(4), 324–339 (1991)
27. Yong-Kui, L.: The generation of straight lines on hexagonal grids. *Comput. Graph. Forum.* 12(1), 21–25 (1993)
28. Yong-Kui, L.: The generation of circular arcs on hexagonal grids. *Comput. Graph. Forum.* 12(1), 27–31 (1993)

Appendix

Proof of Theorem □

The perimeter of a hexagon is P while its area is denoted by A . The lengths of the sides (in order a, b, c, d, e, f) depend on each other by geometry: $a + b = d + e$, $b + c = e + f$, $c + d = f + a$. The proof goes by cases. There are six possible remainder of the division perimeter/6. Further n represents any integer such that the computed perimeter and area are positive ($n \geq 0$ at cases 4 and 5, while $n > 0$ at the other cases).

– case 1: $P = 6n$

(a) Statement to prove: the object is optimal if all sides are equal.

$$P = 6n$$

$$A = 2(2n)(2n) = 6n^2$$

(b) Perimeter is unchanged and some sides are changed so $P = 6n$ still. We can't change the length of only two sides or all six sides because in this case the equations among the sides won't hold.

So we change the length of (at least) two-two sides: ($1 \leq k < n$)

$$n - k, n, n + k, n - k, n, n + k$$

Area:

$$\begin{aligned} A &= 2(n - k + n)(n + k + n - k) - (n - k)^2 - (n - k)^2 = \\ &= 6n^2 - 2k^2 \end{aligned}$$

As $k \geq 1$ this area is smaller than the one in part (a).

– case 2: $P = 6n + 1$

(a) Statement to prove: We get the largest area when the sides are:

$$n - 1, n + 1, n, n, n, n + 1$$

(By symmetry, the area doesn't change when we "move the sides around".)

$$\begin{aligned} A &= 2(n - 1 + n + 1)(n + n) - (n - 1)^2 - n^2 = \\ &= 6n^2 + 2n - 1 \end{aligned}$$

(b) We can change 4 sides so that the equations of the lengths hold.

i. Sides: $n - 1 + k, n + 1 - k, n, n + k, n - k, n + 1$ ($2 \leq k < n, k = 1$ gives no change)

$$P = 6n + 1$$

Area:

$$\begin{aligned} A &= 2(n - 1 + k + n + 1 - k)(n + n + k) - (n - 1 + k)^2 - (n + k)^2 = \\ &= 6n^2 + 2n - 1 + 2k - 2k^2 \end{aligned}$$

We need to show that: $2k - 2k^2 < 0$, it is, iff $k < k^2$.

As $1 < k$ this is always true.

ii. Sides: $n - 1, n + 1 + k, n - k, n, n + k, n + 1 - k$ ($1 \leq k < n$)

$$P = 6n + 1$$

Area:

$$\begin{aligned} A &= 2(n - 1 + n + 1 + k)(n - k + n) - (n - 1)^2 - n^2 = \\ &= 6n^2 + 2n - 1 - 2k^2 \end{aligned}$$

We need to show that: $-2k^2 < 0$, it is, iff $0 < k^2$.

As $k \neq 0$ this is always true.

iii. Sides: $n - 1 - k, n + 1, n + k, n - k, n, n + 1 + k$ ($1 \leq k < n$)

$$P = 6n + 1$$

Area:

$$\begin{aligned} A &= 2(n - 1 - k + n + 1)(n + k + n - k) - (n - 1 - k)^2 - (n - k)^2 = \\ &= 6n^2 + 2n - 1 - 2k - 2k^2 \end{aligned}$$

We need to show that: $-2k - 2k^2 < 0$, it is, iff $k + k^2 > 0$.

As $k \geq 1$ this is always true.

– case 3: $P = 6n + 2$

(a) Statement to prove: We get the largest area when the sides are:

$$n, n, n + 1, n, n, n + 1$$

(The area doesn't change when we "move the sides around".)

$$A = 2(n + n)(n + 1 + n) - n^2 - n^2 = 6n^2 + 4n$$

(b) We can change 4 sides:

i. Sides: $n + k, n, n + 1 - k, n + k, n, n + 1 - k$ ($2 \leq k < n$)

$$P = 6n + 2$$

Area:

$$\begin{aligned} A &= 2(n + k + n)(n + 1 - k + n + k) - (n + k)^2 - (n + k)^2 = \\ &= 6n^2 + 4n + 2k - 2k^2 \end{aligned}$$

We need to show that: $2k - 2k^2 < 0$, it is iff $k < k^2$.

As $1 < k$ this is always true.

- ii. Sides: $n - k, n + k, n + 1, n - k, n + k, n + 1$ ($1 \leq k < n$)

$$P = 6n + 2$$

Area:

$$A = 2(n - k + n + k)(n + 1 + n - k) - (n - k)^2 - (n - k)^2 = 6n^2 + 4n - k^2$$

We need to show that: $-k^2 < 0$, it is, iff $0 < k^2$.

As $k \neq 0$ this is always true.

- iii. Sides: $n, n - k, n + 1 + k, n - k, n + 1 + k$ ($1 \leq k < n$)

$$P = 6n + 2$$

Area:

$$A = 2(n + n - k)(n + 1 + k + n) - n^2 - n^2 = 6n^2 + 4n - 2k - 2k^2$$

We need to show that: $-2k - 2k^2 < 0$, it is, iff $k + k^2 > 0$.

As $0 < k$ this is always true.

– case 4: $P = 6n + 3$

- (a) Statement to prove: We get the largest area when the sides are:

$$n, n + 1, n, n + 1, n, n + 1$$

(The area doesn't change when we "move the sides around".)

$$A = 2(n + n + 1)(n + n + 1) - n^2 - (n + 1)^2 = 6n^2 + 6n + 1$$

- (b) We can change 4 sides so that the equations hold.

- i. Sides: $n + k, n + 1 - k, n, n + 1 + k, n - k, n + 1$ ($1 \leq k < n$)

$$P = 6n + 3$$

Area:

$$A = 2(n + k + n + 1 - k)(n + n + 1 + k) - (n + k)^2 - (n + 1 + k)^2 = 6n^2 + 6n + 1 - 2k^2$$

We need to show that: $-2k^2 < 0$, it is, iff $0 < k^2$.

As $k \neq 0$ this is always true.

- ii. Sides: $n, n + 1 + k, n - k, n + 1, n + k, n + 1 - k$ ($1 \leq k < n$)

$$P = 6n + 3$$

Area:

$$A = 2(n + n + 1 + k)(n - k + n + 1) - n^2 - (n + 1)^2 = 6n^2 + 6n + 1 - 2k^2$$

We need to show that: $-2k^2 < 0$, it is, iff $0 < k^2$.

As $k \neq 0$ this is always true.

- iii. Sides: $n - k, n + 1, n + k, n + 1 - k, n, n + 1 + k$ ($1 \leq k < n$)

$$P = 6n + 3$$

Area:

$$A = 2(n - k + n + 1)(n + k + n + 1 - k) - (n - k)^2 - (n + 1 - k)^2 = 6n^2 + 6n + 1 - 2k^2$$

We need to show that: $-2k^2 < 0$, it is, iff $0 < k^2$.

As $k \neq 0$ this is always true.

– case 5: $P = 6n + 4$

(a) Statement to prove: We get the largest area when the sides are:

$$n, n + 1, n + 1, n + 1, n + 1$$

(The area doesn't change when we "move the sides around".)

$$\begin{aligned} A &= 2(n + n + 1)(n + 1 + n) - n^2 - n^2 = \\ &= 6n^2 + 8n + 2 \end{aligned}$$

(b) We can change 4 sides so that the 3 equations hold.

i. Sides: $n + k, n + 1 - k, n + 1, n + k, n + 1 - k, n + 1$ ($2 \leq k < n, k = 1$ do not change anything)

$$P = 6n + 4$$

Area:

$$\begin{aligned} A &= 2(n + k + n + 1 - k)(n + 1 + n + k) - (n + k)^2 - (n + k)^2 = \\ &= 6n^2 + 8n + 2 + 2k - 2k^2 \end{aligned}$$

We need to show that: $2k - 2k^2 < 0$, it is, iff $k < k^2$

As $1 < k$ this is always true.

ii. Sides: $n, n + 1 + k, n + 1 - k, n, n + 1 + k, n + 1 - k$ ($1 \leq k < n$)

$$P = 6n + 4$$

Area:

$$\begin{aligned} A &= 2(n + n + 1 + k)(n + 1 - k + n) - n^2 - n^2 = \\ &= 6n^2 + 8n + 2 - 2k^2 \end{aligned}$$

We need to show that: $-2k^2 < 0$, it is, iff $0 < k^2$.

As $k \neq 0$ this is always true.

iii. Sides: $n - k, n + 1, n + 1 + k, n - k, n + 1, n + 1 + k$ ($1 \leq k < n$)

$$P = 6n + 4$$

Area:

$$\begin{aligned} A &= 2(n - k + n + 1)(n + 1 + k + n - k) - (n - k)^2 - (n - k)^2 = \\ &= 6n^2 + 8n + 2 - 2k - 2k^2 \end{aligned}$$

We need to show that: $-2k - 2k^2 < 0$, it is, iff $0 < k + k^2$.

It is always true for $k \geq 1$.

– case 6: $P = 6n - 1$

(a) Statement to prove: We get the largest area when the sides are:

$n - 1, n, n, n, n - 1, n + 1$ (The area doesn't change when we "move the sides around".)

$$\begin{aligned} A &= 2(n - 1 + n)(n + n) - (n - 1)^2 - n^2 = \\ &= 6n^2 - 2n - 1 \end{aligned}$$

(b) Again we may change 4 sides so that the equations hold.

i. Sides: $n - 1 + k, n - k, n, n + k, n - 1 - k, n + 1$ ($1 \leq k < n$)

$$P = 6n - 1$$

Area:

$$\begin{aligned} A &= 2(n - 1 + k + n - k)(n + n + k) - (n - 1 + k)^2 - (n + k)^2 = \\ &= 6n^2 - 2n - 1 - 2k^2 \end{aligned}$$

We need to show that: $-2k^2 < 0$, it is, iff $0 < k^2$.

As $1 \leq k$ this is always true.

ii. Sides: $n - 1, n + k, n - k, n, n - 1 + k, n + 1 - k$ ($1 \leq k < n$)

$$P = 6n - 1$$

Area:

$$A = 2(n - 1 + n + k)(n - k + n) - (n - 1)^2 - n^2 =$$

$$= 6n^2 - 2n - 1 - 2k - 2k^2$$

We need to show that: $-2k - 2k^2 < 0$, it is, iff $0 < k + k^2$.

As $k \geq 1$ this is always true.

iii. Sides: $n - k, n + 1, n + 1 + k, n - k, n + 1, n + 1 + k$ ($1 \leq k < n$)

$$P = 6n - 1$$

Area:

$$A = 2(n - 1 - k + n)(n + k + n - k) - (n - 1 - k)^2 - (n - k)^2 =$$

$$= 6n^2 - 2n - 1 - 2k - 2k^2$$

We need to show that: $-2k - 2k^2 < 0$. As $1 \leq k$ this is always true.

All the cases have been analyzed, the proof is finished.

Dynamic Minimum Length Polygon

Jacques-Olivier Lachaud* and Xavier Provençal

Laboratoire de Mathématiques, UMR 5127 CNRS, Université de Savoie,
73376 Le Bourget du Lac, France

{jacques-olivier.lachaud,xavier.provençal}@univ-savoie.fr

Abstract. This paper presents a formal framework for representing all reversible polygonalizations of a digital contour (i.e. the boundary of a digital object). Within these polygonal approximations, a set of local operations is defined with given properties, e.g., decreasing the total length of the polygon or diminishing the number of quadrant changes. We show that, whatever the starting reversible polygonal approximation, iterating these operations leads to a specific polygon: the Minimum Length Polygon. This object is thus the natural representative for the whole class of reversible polygonal approximations of a digital contour. Since all presented operations are local, we obtain the first dynamic algorithm for computing the MLP. This gives us a sublinear time algorithm for computing the MLP of a contour, when the MLP of a slightly different contour is known.

1 Introduction

It is often interesting to construct a polygonal approximation of digital contours (i.e. the boundary of a digital object) in order to study their geometry. We are interested in reversible polygonalizations, which have the property that they are digitized exactly as the input digital contour. Classically, a reversible polygon can be obtained by greedy decomposition of the input contour into longest digital straight segments [21][10]. This decomposition depends on the starting point, but has at most one more edge than the reversible polygon with the minimal number of edges. A reversible polygonal approximation which minimizes the integral summed squared error can also be sought [5]. Another classical reversible polygon is the Minimum Length Polygon (MLP) [12][16][6]. It is a good digital tangent and length estimator [9][3] and is proven to be multigrid convergent in $O(h)$ for digitization of convex shapes, where h is the grid step (reported in [8][17][18]). Several linear-time algorithms exist to compute it [13][14].

The contributions of this paper are twofold. First, we introduce a kind of algebra within all reversible polygonal descriptions of a given digital contour. Each digital contour thus induces a class of reversible polygons which have all the same digitization. A set of valid operations is then defined to pass from one polygon to another within the same class. We show that the subset of operations we provide is necessary and sufficient to go from any element of a class to the

* Partially funded by ANR project FOGRIMMI (ANR-06-MDCA-008-06).

MLP of the same class. In a sense, the MLP acts as a natural representative of all the reversible polygonal approximations. Secondly, this approach leads us to a *dynamic* algorithm for computing the MLP. Since all operations are local with controlled complexity, given the MLP of a given contour C of size n , after a local perturbation on this contour, that is to add or remove one pixel to the region bounded by C , the MLP of this new region can be computed in $O(\log n)$.

Furthermore, a dynamic algorithm for computing the MLP is interesting in several applications. It has been shown that the MLP is a very good regularizer for digital deformable models [22]. Unfortunately, existing algorithms are not dynamic, and the MLP is thus recomputed at each iteration for each possible elementary deformations. A dynamic MLP thus induces a dramatic improvement in the computation speed of digital deformable models. Another application is the computation of multiscale digital representations of a digital object [15]. The MLP of each multiscale contour could thus be computed directly from the decomposition into digital straight segments computed by their analytic approach.

2 Preliminaries

We call *digital contour* C a simple 1-curve in \mathbb{Z}^2 (in the terminology of [8], §7.3.2, p. 243), such that its cell representation has two boundaries that are Jordan curves. The inner curve (resp. outer curve) is called the *inner polygonal curve* of C (resp. the *outer polygonal curve* of C). The *inner polygon* $IP(C)$ is the inner polygonal curve with its inside in \mathbb{R}^2 . Similarly the *outer polygon* $OP(C)$ is the outer polygonal curve with its inside in \mathbb{R}^2 . See Fig. 1 for an illustration. Remark that C , $IP(C)$ and $OP(C)$ are all polyominoes such that $IP(C)$ and $OP(C)$ have vertices in \mathbb{Z}^2 while the interpixel path of C has vertices in $\mathbb{Z}^2 + (1/2, 1/2)$.

One can use for instance a Freeman chain to code a digital contour as a word over the alphabet $(0, 1, 2, 3)$, the associated displacements written as: $\vec{0} = (1, 0)$, $\vec{1} = (0, 1)$, $\vec{2} = (-1, 0)$ and $\vec{3} = (0, -1)$. These words are usually called *contour words* and the contour word of a digital contour C is denoted by $F(C)$. In order to simplify the presentation, we will assume that digital contour are always encoded in a clockwise manner.

Clearly, any curve among the digital contour, the inner polygonal curve and the outer polygonal curve, completely defines the others. For instance starting with the digital contour C , the Freeman chain code of the inner polygonal curve is obtained by removing one step in each turn $ab^k c$ of $F(C)$ with $k \geq 1$ and $(a, b, c) \in \{(0, 3, 2), (1, 0, 3), (2, 1, 0), (3, 2, 1)\}$ by $ab^{k-1}c$ and adding one step in each turn $ab^k c$ with $k \geq 1$ and $(a, b, c) \in \{(0, 1, 2), (1, 2, 3), (2, 3, 0), (3, 0, 1)\}$ by $ab^{k+1}c$. Similarly, the Freeman chain code of the outer polygonal curve is obtained from $F(C)$ in the same way by adding one step in former case and removing one in the latter one.

2.1 Minimum Length Polygon

Following the works of Sloboda, Zafko and Stoer [17,20,19] (or see [7,8]), we define the minimum length polygon (MLP) of C as the shortest Jordan curve

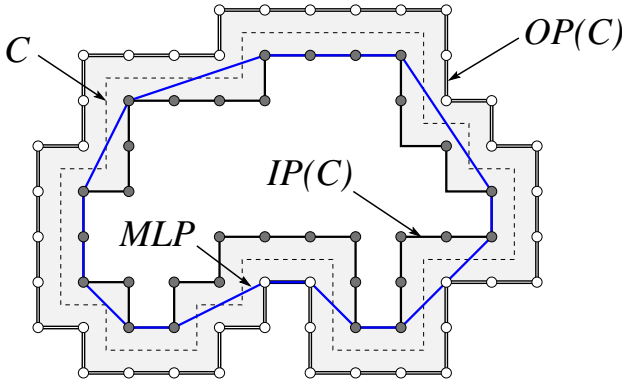


Fig. 1. A digital contour C with its inner polygon $IP(C)$, its outer polygon $OP(C)$ and its MLP which is, in a clockwise manner, up to circular permutation, the grid-curve $[(1, 0)^3, \sigma^+, (3, 2), (1, 0), \sigma^+, \widetilde{(1, 1)}, \widetilde{(1, 1)}, (1, 0), \sigma^+, \widetilde{(1, 1)}, (1, 0), \sigma^+, \widetilde{(2, 1)}, (1, 0), \sigma^+, (1, 1), (1, 0)^2, \sigma^+, (1, 2), (3, 1)]$

which stays inside the 1-pixel width band drawn by the cell representation of C . More precisely, letting \mathcal{A} be the family of simply connected compact sets of \mathbb{R}^2 , we define:

Definition 1. The minimum perimeter polygon of two polygons V, U with $V \subset U^\circ \subset \mathbb{R}^2$ is a subset P of \mathbb{R}^2 such that

$$P = \operatorname{argmin}_{A \in \mathcal{A}, V \subseteq A, \partial A \subset U \setminus V^\circ} \operatorname{Per}(A), \tag{1}$$

where $\operatorname{Per}(A)$ stands for the perimeter of A , more precisely the 1-dimensional Hausdorff measure of the boundary of A .

Definition 2. The minimum length polygon (MLP) of a digital contour C is the minimum perimeter polygon of $IP(C), OP(C)$.

Other equivalent definitions for MLP may be found in [13,14]: they are based either on arithmetic or word combinatorics. In [17], it is shown that Equation (1) has a unique solution.

3 Algebra on Reversible Polygonal Representations

We wish to classify polygons with integer vertices according to the digital contour that they represent. Indeed, many different polygons may represent the same contour. More precisely, a reversible polygonal representation (RPR) of C is a polygon whose edges stays in $OP(C) \setminus IP(C)^\circ$, whose vertices are integer vertices of either $OP(C)$ or $IP(C)$ and with an extra bit of information per vertex specifying if it touches the inside or the outside of the band. It is clear that the

digital contour can be reconstructed from such object by computing the set of intersected pixels. If an edge intersects the interior of a pixel, then it is part of C while if the edges intersect only the border of a pixel, to the abovementioned extra bit of information allows to determine if this pixel is part of C or not.

We wish now to find a natural representative for all the RPR of a given contour C . Our approach is to look for the shortest one. Since the MLP of C is one RPR of C , our unique representative will be the MLP. We thus provide a notation for RPR and a set of operations within RPR. These operations are designed so that they shorten the RPR and they act locally. Furthermore, we show that they preserve the digital contour.

3.1 Grid-Vector, Grid-Curve

Definition 3. A grid-vector is a triplet $x = ((p, q), k, \delta_x) \in \mathbb{N}^2 \times \mathbb{N} \times \mathbb{B}$ where $\gcd(p, q) = 1$, q/p is the slope of x (with the convention that $\infty = 1/0$), $k \geq 1$ is its number of repetitions and the boolean δ_x is true when one endpoint of x lies on the inner polygonal curve and the other endpoint lies on the outer polygonal curve.

Such grid-vector $x = ((p, q), k, \delta_x)$ is noted $(p, q)^k$ if δ_x is F and $\widetilde{(p, q)^k}$ if δ_x is T, the \sim meaning that a side change has occurred. A grid-vector has no specific orientation since its slope is in $[0, \infty[$. Any grid-vector with slope 0 or ∞ is called *trivial*. The translation associated to a grid-vector is always given relatively to a pair of letter (a, b) from the alphabet $\{0, 1, 2, 3\}$ and is denote using $\xrightarrow{(a,b)}$. These letters represents the current orientation context, i.e. the oriented quadrant. The symbol \sim specifies a side change and thus inverts the current orientation context. Therefore, after this edge, the context is changed (see Fig. 2). Translations for arbitrary grid vectors are given by the formulae:

$$\xrightarrow{(a,b)}_{(p,q)^k} = k(p\vec{a} + q\vec{b}), \quad \text{and} \quad \xrightarrow{\widetilde{(a,b)}}_{(p,q)^k} = k(p\vec{b} + q\vec{a}).$$

The reversed point of view induced by the symbol \sim explains the different formulae for translation.

Of course, such translation loses any geometrical interpretation when the pair of letters (a, b) represent elementary steps in the opposite direction as in the case of $(0, 2)$ or $(1, 3)$. This situation should never happen.

In order to completely describe a circular band, grid-vectors alone are not enough. We need extra information regarding quadrant changes. That is why we introduce the σ operators that act solely on the current orientation context (i.e., on the pairs of letters):

$$\begin{aligned} \sigma^-(a, b) &= (b, \bar{a}): \text{ this operator is a turn toward the exterior,} \\ \sigma^+(a, b) &= (\bar{b}, a): \text{ this operator is a turn toward the interior,} \\ &\text{with the convention } \bar{0} = 2, \bar{1} = 3, \bar{2} = 0, \bar{3} = 1. \end{aligned}$$

In order to unify the notations, we associate the nil vector to any operator so that $\overrightarrow{\sigma^+} = \overrightarrow{\sigma^-} = \overrightarrow{(0, 0)}$. Similarly, it is convenient to see each grid-vector as an operator on a pair of letters acting in the following way, let $x = ((p, q), k, \delta_x)$,

$$x(a, b) = \begin{cases} (b, a) & \text{if } \delta_x \text{ is } \mathbb{T}, \\ (a, b) & \text{otherwise.} \end{cases}$$

An ordered list of grid-vectors and operators defines a 1-pixel width band in the following way.

Definition 4. A grid-curve Γ is a list $\Gamma = [l_0, l_1, \dots, l_{n-1}]$ where each l_i is either a grid-vector or one of the operators σ^+, σ^- .

The geometrical interpretation of a grid-curve $\Gamma = [l_0, l_1, \dots, l_{n-1}]$ starting with the pair of letters (a, b) is the polygonal curve $P_\Gamma = [P_0, P_1, \dots, P_n]$ where each point P_i is given by:

- $P_0 = (0, 0)$ and $(a_0, b_0) = (a, b)$.
- $P_{i+1} = P_i + \overrightarrow{\overset{(a_i, b_i)}{l_i}}$ and $(a_{i+1}, b_{i+1}) = l_i(a_i, b_i)$ for each $i \in \{0, 1, \dots, n - 1\}$.

Figure 2 illustrates this construction. This figure also highlights the fact that each grid-curve defines a one pixel-wide band bounded on each side by a digital curve and that the segment associated with each grid-vector $x = ((p, q), k, \delta_x)$ intersects one of these digital curves exactly $k + 1$ times. More precisely, each segment starts on one of the two digital curves that form the band, and it intersects one curve exactly every $p + q$ steps. In order to simplify the notations when concatenating two grid-curves, we write $\Gamma(a, b) = (a_n, b_n)$.

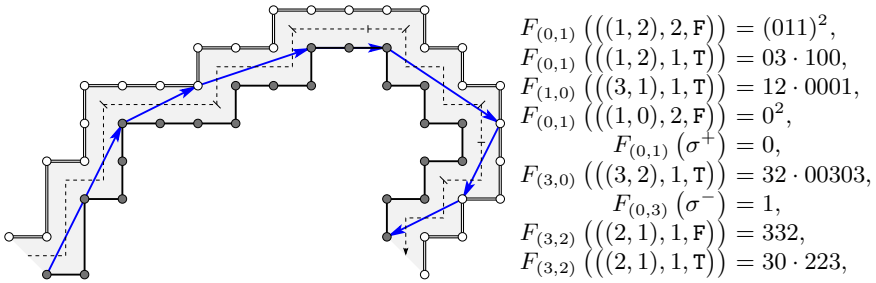
3.2 Christoffel Words, Interpixel Path, RPR

In order to digitize a grid-vector as a piece of digital contour, we make use of a well-known discrete analog to straight segments, the Christoffel words [2, 11]. They have a lot of equivalent definitions, for instance they are the 4-connected Freeman chaincode between two consecutive upper leaning points of a digital straight line, or they are the Lyndon factors of sturmian words (e.g., see [1, 13]).

The Christoffel word of slope q/p over the alphabet (a, b) is denoted by $\mathcal{C}_{q/p}^{(a,b)}$. We then associate to each element l_i of a grid-curve $\Gamma = [l_0, l_1, \dots, l_{n-1}]$ a word $F_{(a,b)}(l_i)$ defined as

$$\begin{aligned} F_{(a,b)}((p, q)^k) &= \left(\mathcal{C}_{q/p}^{(a,b)}\right)^n, & F_{(a,b)}(\sigma^-) &= \bar{b}, \\ F_{(a,b)}(\overline{(p, q)^k}) &= a\bar{b}F_{(b,a)}((p, q)^k), & F_{(a,b)}(\sigma^+) &= a. \end{aligned}$$

The word $F_{(a,b)}(\Gamma)$ is defined by gluing all $F(l_i)$. It may contain factors of the form $a\bar{a}$ which have no real geometrical interpretation. We thus project this word in the free group in order to remove these back and forth moves. This reduced word (each $a\bar{a} = \varepsilon$, the empty word) is called the *interpixel path* $F_{(a,b)}^\varepsilon(\Gamma)$. See Fig. 2.



$$\begin{aligned}
 w &= 011011 \cdot 03 \cancel{100} \cdot 12 \cancel{001} \cdot 00 \cdot 0 \cdot 32 \cancel{030} \cancel{31} 332 \cdot 3 \cancel{02} 23, \\
 &= 011011 \cdot 000 \cdot 1001 \cdot 00 \cdot 0 \cdot 3030 \cdot 3 \cdot 32 \cdot 323.
 \end{aligned}$$

Fig. 2. Illustration of the grid-curve $\Gamma = [(1, 2)^2, \widetilde{(1, 2)}, \widetilde{(3, 1)}, (1, 0)^2, \sigma^+, \widetilde{(3, 2)}, \sigma^-, (2, 1), \widetilde{(2, 1)}]$ starting with the letters (0, 1). Each vector of the polygonal curve P_Γ is represented by an arrow. The interpixel path w obtained by the concatenation of the Freeman code associated to each element of the grid-curve Γ is shown with a dashed line.

Now, let (Q_i) be the clockwise vertices of some RPR of C and let λ_i be the boolean that is true when Q_i lies on $IP(C)$. The following sequence defines the grid-curve of Q , for all i :

1. if $\overrightarrow{Q_{i-1}Q_i}$ and $\overrightarrow{Q_iQ_{i+1}}$ are not in the same quadrant:
 - if $\lambda_i = T$
 - if $\overrightarrow{Q_iQ_{i+1}}$ is to the right of $\overrightarrow{Q_{i-1}Q_i}$, append as many σ^+ as the number of clockwise $\pi/2$ rotations to bring these two vectors in the same quadrant.
 - else append as many σ^- as the number of counterclockwise $\pi/2$ rotations to bring these two vectors in the same quadrant.
 - else append the opposite operators of above paragraph.
2. letting (a, b) be the current orientation context, then $\overrightarrow{Q_iQ_{i+1}}$ is some $u\vec{a} + v\vec{b}$; let also $p = \text{gcd}(u, v)$.
 - if $\lambda_i = \lambda_{i+1}$ append $(u/p, v/p)^p$,
 - else append $(v/p, u/p)^p$.

The following proposition formalizes the fact that the previous construction does build the correct digital contour.

Proposition 1. *For any RPR Q of a digital contour C , the interpixel path of the grid-curve of Q has the same chaincode as C up to conjugacy.*

For space reasons, we do not detail the proof here. Using this property we will be able to validate simplification operations defined on grid-curves by showing that they preserve the digital contour.

In order to adopt a local approach to process RPR and the associated grid-curve, we will consider sublists of such grid-curves. Note that they are also grid-curves (but they do have extremities unlike grid-curves defined by RPR).

We introduce an equivalence relation between these objects, because they geometrically define the same one pixel-wide band and preserve the orientation context.

Definition 5. *Two grid-curves $\Gamma = [l_0, l_1, \dots, l_{n-1}]$ and $\Gamma' = [l'_0, l'_1, \dots, l'_{m-1}]$ are equivalent, noted $\Gamma \equiv \Gamma'$, if $F_{(0,1)}^\varepsilon(\Gamma) = F_{(0,1)}^\varepsilon(\Gamma')$ and $\Gamma(0, 1) = \Gamma'(0, 1)$.*

For instance, $[(1, 1), (1, 1)] \equiv [(1, 1)^2]$ since $01 \cdot 01 = (01)^2$ and both curves leave the alphabet unchanged. On the other hand, Fig. 5 shows a less trivial example.

4 Simplification Rules

In order to compute the canonical representation of a grid path, we define simplification rules, which are detailed in the following subsections:

- Merging rules: given by equations (2) and (3).
- Splitting rules: given by equation (4).
- Operator simplification rules: given in Section 4.4.

When applied to a grid curve Γ , each of these rules generates another grid curve Γ' such that $F_{(a,b)}^\varepsilon(\Gamma) = F_{(a,b)}^\varepsilon(\Gamma')$ but in each case, we can ensure that $\|\Gamma\| < \|\Gamma'\|$ (the euclidean length is smaller), or $\|\Gamma\| = \|\Gamma'\|$ and $|F_{(a,b)}(\Gamma)| < |F_{(a,b)}(\Gamma')|$ (same euclidean length but shorter word).

4.1 Grid-Vectors Fusion Rules

Given two vectors $\vec{u} = (p, q)$ and $\vec{v} = (r, s)$ it is well know that the area of the oriented parallelogram defined by \vec{u} and \vec{v} is simply $ps - qr$. According to this, we define the product of $x = ((p, q), k, \delta_x)$ and $y = ((r, s), l, \delta_y)$ as

$$x \otimes y = \begin{cases} ps - qr & \text{if } \delta_y = \mathbf{F}, \\ pr - qs & \text{if } \delta_y = \mathbf{T}. \end{cases}$$

The sign, positive or negative, of $x \otimes y$ has the following geometrical interpretation:

- $x \otimes y < 0$. In such case, the grid-curve $[x, y]$ defines a convex vertex of the RPR that is optimal in the sense that it may not be replaced by a shorter polygonal curve that stays within $\text{OP}(C) \setminus \text{IP}(C)^\circ$.
- $x \otimes y = 0$. In such case, x and y are co-linear. When $\delta_y = \mathbf{F}$ then y may simply be added to the number of repetitions of x so that $[x, y]$ may be replaced by $[((p, q), k + l, \delta_x)]$:

$$[((p, q), k, \delta_x), ((p, q), l, \mathbf{F})] \equiv [((p, q), k + l, \delta_x)]. \tag{2}$$

On the other hand, if $\delta_y = \mathbf{T}$ then no simplification is possible and $[x, y]$ is optimal.

- $x \otimes y > 0$. In such case, $[x, y]$ is not optimal and there exist some grid-curve Γ such that $\|\Gamma\| < \|[x, y]\|$.

Therefore we focus our attention on pairs of grid-vectors x and y such that $x \otimes y > 0$. There are two cases to consider, whether $x \otimes y = 1$ or $x \otimes y > 1$. In the first case, the two grid-vectors are compatible: there is no integer points in the triangle formed by (p, q) and (r, s) and these two segments may be replaced by the segment $(p + r, q + s)$. This is detailed in Section 4.2. In the second case however, the latter segment would go outside of $OP(C) \setminus IP(C)^\circ$. Hence, $[x, y]$ has to be replaced by some grid-curve $\Gamma = [l_1, l_2, \dots, l_n]$ in which for each i from 1 to $n - 1$, $l_i \otimes l_{i+1} \leq 0$. In section 4.3 we show how to compute Γ in time proportional to the depth of the continued fraction development of the slopes of x and y .

4.2 Merging Grid-Vectors

The following merging operation is based on the well known *splitting formula* of digital straight segments (see 23). This is equivalently known in the field of word combinatorics as the *standard factorization* of Christoffel words (see 112).

Let $x = ((p, q), 1, \mathbf{F})$ and $y = ((r, s), 1, \mathbf{F})$ be two grid-vectors such that $x \otimes y = 1$. In such case, x and y may be merged in order to form the grid-vector $z = ((p + r, q + s), 1, \mathbf{F})$ so that

$$[x, y] \equiv [z] \text{ and } \|[x, y]\| > \|[z]\|.$$

In particular, a grid-curve of the form $[x, x, \dots, x, y]$ with k copies of x may be simplified to the shorter $[((kp + r, kq + s), 1, \delta_x)]$. Similarly, $[x, y, y, \dots, y]$, with l copies of y , is merged to $[((p + lr, q + ls), 1, \mathbf{F})]$. On the other hand, if both x and y are repeated more than one time, a more complex simplification operation (explained in Section 4.3) is needed. By taking into account the possible changes from the inner to the outer polygon and vice versa, we obtain the following merging rule:

- Let $x = ((p, q), k, \delta_x)$ and $y = ((r, s), l, \delta_y)$ with either $\delta_y = \mathbf{F}$ and $\min(k, l) = 1$ or $\delta_y = \mathbf{T}$ and $l = 1$, then

$$[x, y] \equiv [z] \text{ where } z = \begin{cases} ((kp + lr, kq + ls), 1, \delta_x) & \text{if } \delta_y = \mathbf{F}. \\ ((kp + ls, kq + lr), 1, \neg\delta_x) & \text{otherwise.} \end{cases} \quad (3)$$

4.3 Split and Merge Formulae

In order to simplify a grid-curve into a shorter one, we introduce splitting operations. Given two grid-vectors x and y such that $x \otimes y > 1$, both x and y are split in a specific manner. The resulting grid-curve is such that merging operations are sufficient in order to obtain the shortest equivalent grid-curve.

Let $x = ((p, q), 1, \mathbf{F})$ be a non-trivial grid-vector and let $[u_0; u_1, \dots, u_n]$ be the continued fraction development of q/p . We note q_i/p_i is the i -th convergent of q/p



Fig. 3. Illustration of the upper and lower splitting operations. On the left, $(3/5)$ is split to $S_{\downarrow}((3/5)) = [(1/2)^2, (1/1)]$ and $S^{\uparrow}((3/5)) = [(1/1), (2/1), \widetilde{(1/2)}]$. On the right, the segment $(5/3)$ is split to $S_{\downarrow}((5/3)) = [(1/2)^2, \widetilde{(1/1)}]$ and $S^{\uparrow}((5/3)) = [\widetilde{(1/1)}, (2/1)^2]$. On both examples, the upper splitting S^{\uparrow} is shown in red while the lower splitting S_{\downarrow} is shown in green.

that is the fraction $q_i/p_i = [u_0; u_1, \dots, u_i]$. In order to lighten the presentation, we introduce the following notations: $x_i = (p_i, q_i)$, $x_{-1} = (0, 1)$, $x_{-2} = (1, 0)$. Also, more generally, if $y = ((r, s), l, \delta_y)$ then $y^{-1} = ((s, r), l, \delta_y)$ and $\widetilde{y} = ((r, s), l, -\delta_y)$. The following operation is exactly the opposite operation of (3).

Definition 6. The basic splitting of the grid-vector x_n is the grid-curve:

$$S(x_n) = \begin{cases} [x_{2m-2}, x_{2m-1}^{u_{2m}}] & \text{if } n = 2m, \\ [x_{2m}^{u_{2m+1}}, x_{2m-1}] & \text{if } n = 2m+1, \end{cases}$$

Computing the inverse of any number from its continued fraction development is an easy task. This observation leads to following formula:

$$\text{if } S(x) = [u^k, v^l] \text{ then } S(x^{-1}) = [(v^{-1})^l, (u^{-1})^k].$$

The basic splitting operation is only defined on grid-vector of the form $x = ((p, q), 1, \delta_x)$ with $\delta_x = F$. We extend it to all segments of multiplicity one.

Definition 7. The upper splitting S^{\uparrow} and the lower splitting S_{\downarrow} are defined as follow, let $[u^k, v^l] = S((p, q), 1, F)$ then

	$x = ((p, q), 1, F)$	$x = ((q, p), 1, T)$
$S^{\uparrow}(x)$	$[(\widetilde{(v^{-1})^l}, (u^{-1})^{k-1}, \widetilde{u})]$	$[(\widetilde{(v^{-1})^l}, (u^{-1})^k)]$
$S_{\downarrow}(x)$	$[u^k, v^l]$	$[u^k, v^{l-1}, \widetilde{v^{-1}}]$

By repetition of the above splittings, any grid-vector may be decomposed in order to isolate a trivial grid-vector on the left or the right. See Fig. 4 for an illustration of the following definition.

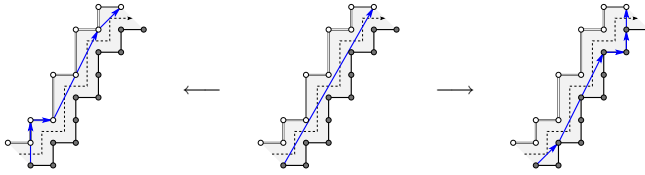


Fig. 4. In the center, an illustration of the grid-vector $x = \widetilde{(7, 4)}$. On the left, the left splitting $S_-(x) = \widetilde{[(1, 0)^2, (0, 1), (2, 1)^2, (1, 1)]}$. On the right, the right splitting $S_+(x) = \widetilde{[(1, 1), (1, 2)^2, (1, 0), (0, 1), (1, 0)]}$.

Definition 8. Let $x = ((p, q), k, \delta_x)$ be a grid-vector, the left splitting $S_-(x)$ and the right splitting are defined as follows:

- if $k = 1$ and x is trivial, $S_-(x) = S_+(x) = [x]$;
 - if $k \geq 2$, $S_-(x) = S_-(((p, q), 1, \delta_x)) + [((p, q), k - 1, F)]$,
 $S_+(x) = [((p, q), k - 1, \delta_x)] + S_+((p, q), 1, F)$;
 - otherwise, let $S^\uparrow(x) = [l_1, l_2, \dots, l_n]$ and $S_1(x) = [l'_1, \dots, l'_m]$ (4)
- $$S_-(x) = S_-(l_1) + [l_2, \dots, l_n],$$
- $$S_+(x) = \begin{cases} [l_1, \dots, l_{n-1}] + S_-(l_n) & \text{if } \delta_x = F, \\ [l'_1, \dots, l'_{m-1}] + S_+(l'_m) & \text{if } \delta_x = T. \end{cases}$$

Proposition 2. Let $x = ((p, q), k, \delta_x)$ be a grid-vector, both left and right splittings of x are such that: $[x] \equiv [S_-(x)] \equiv [S_+(x)]$.

Sketch of the proof. These equivalences come from successive applications of splitting formula on digital straight segments.

Proposition 3. The number of grid-vector in both left and right splittings of $x = ((p, q), k, \delta_x)$ is $\Theta(n)$ where n is the depth of the continued fraction development of q/p .

Proof. It suffices to see that each time a basic splitting operation is performed (Definition 6), the depth of the continued fraction development of the slopes decreases by one or two. □

Since the depth n of a continued fraction $q/p = [u_0; u_1, \dots, u_n]$ is smaller than $\log_2(p + q)$, a weaker form of the previous proposition states that the number of grid-vector is some $O(\log N)$ where N is the length of the contour word.

Algorithm 1 illustrates how to simplify a grid-curve by the use of our *split and merge* formulae. The function `Merge` called on line 15 of Algorithm 1 performs the following task: the two grid-curves given as input being a right splitting, $S_-(x) = [r_1, r_2, \dots, r_n]$ and a left splitting, $S_-(y) = [l_1, l_2, \dots, l_m]$, both r_n

Algorithm 1. Simplification

```

Input:  $\Gamma = [l_0, l_1, \dots, l_{n-1}]$  where each  $l_i$  is a grid-vector
1  $\Delta = []$ ;
2 while  $\Gamma$  is not empty do
3    $y \leftarrow \Gamma.pop\_front()$  ;
4   if  $\Delta$  is empty then
5      $\Delta.push\_back(y)$ ;
6   else
7      $x \leftarrow \Delta.pop\_back()$ ;
8     if  $x \otimes y < 0$  or  $(x \otimes y = 0$  and  $\delta_y = T)$  then
9        $\Delta.push\_back(x)$ ;
10       $\Delta.push\_back(y)$ ;
11     else
12       if there exist  $z$  such that  $[z] \equiv [x, y]$  then
13          $\Gamma.push\_front(z)$ ;
14       else
15          $\Gamma \leftarrow \text{Merge}(S_{\rightarrow}(x), S_{\leftarrow}(y)) + \Gamma$ ;
16 return  $\Delta$ ;
```

and l_1 are trivial and may be replaced by the grid-vector $(1, 1)$. Both lists are then concatenated into $C = [r_1, \dots, r_{n-1}, (1, 1), l_2, \dots, l_m]$. Finally, if there is a pair of consecutive grid-vectors u, v in C such that, according to the fusion rules described by equations (2) and (3), there exist z satisfying $[u, v] \equiv [z]$, then the pair u, v is replaced by z . This last step is performed iteratively until there are no such pairs left.

The following proposition states that whenever line 15 of Algorithm 1 is executed, the grid-curve is simplified in the sense that output curve is strictly shorter.

Proposition 4. *Given two grid-vectors x and y such that $x \otimes y > 1$, the grid-curve $\Gamma = \text{Merge}(S_{\rightarrow}(x), S_{\leftarrow}(y))$ is such that $\|\Gamma\| < \|[x, y]\|$.*

Sketch of the proof. Consider the grid-curve $\Gamma = S_{\rightarrow}(x) + S_{\leftarrow}(y)$ and the associated polygon $P_{\Gamma} = [P_0, P_1, \dots, P_n]$, as defined in Section 3.1. Let P_i be the point between $S_{\rightarrow}(x)$ and $S_{\leftarrow}(y)$. One checks that all points $P_1, P_2, \dots, P_{i-1}, P_{i+1}, \dots, P_{n-1}$ lies on the same polygonal contour (inner or outer) while P_i lies on the other one. The fusion of two grid-vectors removes a points from the associated polygon. In particular, the first operation performed by **Fusion** is to remove P_i from P_{Γ} . Each pair of consecutive grid-vectors u, v from C is such that either $u \otimes v \leq 0$ or the pair u, v is replaced by a single grid-vector. When the process stops, the resulting grid-curve Δ defines a convex region.

Finally, let X be some point far enough from P_0 in the direction $\vec{y} - \vec{x}$. Consider Π_{Δ} the polygon defined by Δ followed by the line segments $\overline{P_n X}$ and $\overline{X P_0}$ and, similarly, Π_{xy} the polygon $P_0 P_i P_n A$. The polygon Π_{Δ} is a convex polygon strictly included in Π_{xy} and thus its perimeter is strictly smaller.

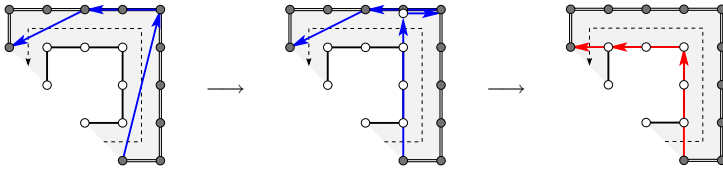


Fig. 5. Illustration of a grid-curve simplification. On the left $\Gamma = [(1, 4), \sigma^-, (0, 1)^2, \sigma^-, (2, 1)]$ and on the right $\Delta = [(\widetilde{1, 0})^3, \sigma^+, \sigma^+, (0, 1)^2, (\widetilde{1, 0})]$ and $\Gamma \equiv \Delta$ since $01111 \cdot 3 \cdot 0 \cdot 22223 = 03 \cdot (1)^3 \cdot 1 \cdot 2 \cdot (2)^2 \cdot 30 \cdot 2$ and both curves transform the alphabet $(0, 1)$ into $(2, 3)$. On the other hand the euclidean length of Δ is smaller than the length of Γ .

4.4 Simplification Rules for Operators

As mentioned previously, operator σ^+ codes a quadrant change toward the inside which means that a part of a RPR of the form $[x, \sigma^+, y]$ is locally optimal. On the other hand, operator σ^- may not appear in a MLP since it codes a quadrant change toward the outside (See Fig. 2). We define local rules to remove the σ^- operators in a grid-curve.

First of all, using the left and right splitting operations defined in the previous section, we can easily update the grid-curve so that around a σ^- operator there are only trivial grid-vectors. Also, by using the relation $[(0, 1)] \equiv [\sigma^-, (1, 0), \sigma^+]$ we may only consider trivial grid-vectors with slope 0.

Simplification rules for operator σ^- are all local and thus treated in constant time. These rules are of three types:

Push to the right. The following rules create a shorter grid-curve by replacing a pattern of the form $_ \uparrow$ by \nearrow .

$$[(1, 0), \sigma^-, (1, 0)] \equiv [(1, 1), \sigma^-], \quad \text{and} \quad [(\widetilde{1, 0}), \sigma^-, (1, 0)] \equiv [(\widetilde{1, 1}), \sigma^-]. \quad (5)$$

Cancellation rules. Given an occurrence of σ^- in a grid-curve, the trivial grid-vectors right before and after may go back and forth within a single pixel. Such situation appears in a *locally-closed pattern*. It is a grid-curve such that: (i) it includes exactly one or two trivial grid-vectors before σ^- and one or two more after σ^- , (ii) it is closed in the sense that the first and last points of P_Γ are the same. Given such locally-closed pattern Γ , if there exists $\Delta \in \{[], [\sigma^+], [\sigma^+, \sigma^+]\}$ such that $\Gamma \equiv \Delta$ then replace Γ by Δ . When implemented these rules may be tabulated so as to apply them in constant time.

Correction rules. When performing a left or right splitting operation, we make the assumption that all points below the grid-vector belongs to the same polygon (inner or outer) and all those above lies in the other polygon. Although this is true in general, it may not be the case for extremities. For example, consider the grid-vector $(1, 4)$ from Fig. 5. In order to obtain a trivial grid-vector right before σ^- , $(1, 4)$ is replaced by $S_{\rightarrow}(x) = [(\widetilde{1, 0})^4, (\widetilde{1, 0})]$ while the correct substitution

would be to replace $(1, 4)$ by $[(\widetilde{1, 0})^3, (\widetilde{0, 1}), (1, 0)]$. One could modify the splitting operations in order to take these situation into account but we prefer to use above splitting operations as is and eventually correct the curve afterward. These errors are locally-closed patterns in which the curve changes from one polygon (inner or outer) to the other one an odd number of times. Geometrically, this would imply that a point belongs to both polygons at the same time, which is impossible. Our splitting operations may cause only two types of these faulty locally-closed patterns. Let $\Delta = [l_1, \dots, l_k]$ be a locally-closed pattern of the grid-curve Γ :

- if $F_{(0,1)}^\varepsilon(\Delta) = 0$ and $\Delta((0, 1)) = (2, 1)$. In this case, let x be the trivial grid-vector right after Δ in Γ , the pattern $\Delta + [x] = [l_1, \dots, l_k, x]$ is replaced by $[\sigma^-, \tilde{x}]$.
- if $F_{(0,1)}^\varepsilon(\Delta) = 0$ and $\Delta((0, 1)) = (3, 2)$. In this case, let y be the trivial grid-vector with multiplicity one right before Δ in Γ , the pattern $[y] + \Delta$ is replaced by $[\sigma^+, \tilde{y}, \sigma^-]$.

5 Concluding Remarks

We have presented three types of simplification rules (merging rules, splitting rules and operators simplification rules) that allows to compute the MLP of any given RPR with local operations. Starting from the interpixel path of some discrete region, one may use these rules in order to compute its MLP. On the other hand, the overall computation time would be significantly lower by using the algorithms presented in [13,14] since their approach are more straightforward than the iterative one obtained from our local operations. Nevertheless, given some discrete contour C and its MLP, if a local perturbation is performed on C , for instance change a factor 01 by 10 in the contour word, a RPR which is not the MLP can be deduced directly from the previous one. Using the techniques presented in this paper, a dynamic computation of the new MLP from this RPR is possible in time sublinear with respect to the length of the section over which both MLPs differ. We plan to use this algorithm in digital deformable partition models where the length of region boundaries have to be computed after each modification [22,4].

References

1. Berstel, J., Lauve, A., Reutenauer, C., Saliola, F.V.: Combinatorics on words, CRM Monograph Series, vol. 27. American Mathematical Society, Providence (2009); christoffel words and repetitions in words
2. Borel, J.P., Laubie, F.: Quelques mots sur la droite projective réelle. J. Théor. Nombres Bordeaux 5(1), 23–51 (1993)
3. Coeurjolly, D., Klette, R.: A comparative evaluation of length estimators of digital curves. IEEE Trans. Pattern Analysis and Machine Intelligence 26(2), 252–258 (2004)
4. Damiand, G., Dupas, A., Lachaud, J.O.: Combining topological maps, multi-label simple points and minimum length polygon for efficient digital partition model (2011), submitted to IWCIA (2011)

5. Feschet, F.: Fast guaranteed polygonal approximations of closed digital curves. In: Kalviainen, H., Parkkinen, J., Kaarna, A. (eds.) SCIA 2005. LNCS, vol. 3540, pp. 910–919. Springer, Heidelberg (2005)
6. Hobby, J.D.: Polygonal approximations that minimize the number of inflections. In: Proc. SODA, pp. 93–102 (1993)
7. Klette, R., Kovalevsky, V., Yip, B.: On length estimation of digital curves. In: Vision Geometry VIII, vol. 3811, pp. 117–128 (1999)
8. Klette, R., Rosenfeld, A.: Digital Geometry - Geometric Methods for Digital Picture Analysis. Morgan Kaufmann, San Francisco (2004)
9. Klette, R., Yip, B.: The length of digital curves. *Machine Graphics Vision* 9(3), 673–703 (2000)
10. Lindenbaum, M., Brückstein, A.: On recursive, $O(N)$ partitioning of a digitized curve into digital straight segments. *IEEE Trans. On Pattern Analysis and Machine Intelligence* 15(9), 949–953 (1993)
11. Lothaire, M.: Algebraic combinatorics on words, *Encyclopedia of Mathematics and its Applications*, vol. 90. Cambridge University Press, Cambridge (2002)
12. Montanari, U.: A note on minimal length polygonal approximation to a digitized contour. *Communications of the ACM* 13(1), 41–47 (1970)
13. Provençal, X., Lachaud, J.-O.: Two Linear-Time Algorithms for Computing the Minimum Length Polygon of a Digital Contour. In: Brlek, S., Reutenauer, C., Provençal, X. (eds.) DGCI 2009. LNCS, vol. 5810, pp. 104–117. Springer, Heidelberg (2009)
14. Roussillon, T., Tougne, L., Sivignon, I.: What does digital straightness tell about digital convexity? In: Wiederhold, P., Barneva, R.P. (eds.) IWCIA 2009. LNCS, vol. 5852, pp. 43–55. Springer, Heidelberg (2009)
15. 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)
16. Sklansky, J., Chazin, R.L., Hansen, B.J.: Minimum perimeter polygons of digitized silhouettes. *IEEE Trans. Computers* 21(3), 260–268 (1972)
17. Sloboda, F., Stoer, J.: On piecewise linear approximation of planar Jordan curves. *J. Comput. Appl. Math.* 55(3), 369–383 (1994)
18. Sloboda, F., Zafko, B.: On one-dimensional grid continua in \mathbb{R}^2 . Tech. rep., Institute of Control Theory and Robotics, Slovak Academy of Sciences, Bratislava, Slovakia (1996)
19. Sloboda, F., Zafko, B.: On approximation of jordan surfaces in 3D. In: Bertrand, G., Imiya, A., Klette, R. (eds.) Digital and Image Geometry. LNCS, vol. 2243, pp. 365–386. Springer, Heidelberg (2002)
20. Sloboda, F., Zafko, B., Stoer, J.: On approximation of planar one-dimensional continua. In: Klette, R., Rosenfeld, A., Sloboda, F. (eds.) Advances in Digital and Computational Geometry, pp. 113–160 (1998)
21. Smeulders, A.W.M., Dorst, L.: Decomposition of discrete curves into piecewise straight segments in linear time. In: Melter, R.A., Rosenfeld, A., Bhattacharya, P. (eds.) Vision Geometry. Contemporary Mathematics, vol. 119, pp. 169–195. Am. Math. Soc., Providence (1991)
22. de Vieilleville, F., Lachaud, J.-O.: Digital deformable model simulating active contours. In: Brlek, S., Reutenauer, C., Provençal, X. (eds.) DGCI 2009. LNCS, vol. 5810, pp. 203–216. Springer, Heidelberg (2009)
23. Voss, K.: Discrete images, objects, and functions in \mathbf{Z}^n , *Algorithms and Combinatorics*, vol. 11. Springer, Berlin (1993)

On Some Classes of 2D Languages and Their Relations

Marcello M. Bersani, Achille Frigeri, and Alessandra Cherubini

Politecnico di Milano

bersani@elet.polimi.it, {achille.frigeri,alessandra.cherubini}@polimi.it

Abstract. Many formal models have been proposed to recognize or to generate two-dimensional words. In this paper, we focus our analysis on (regular) pure 2D context-free grammars, regional tile grammars and Průša grammars, showing that nevertheless they have been proposed as a generalization of string context free grammars their expressiveness is different. This work refines the relationship among the classes of languages generated by the above grammars and Local languages and states some considerations about closure properties of (regular) pure 2D context-free languages.

1 Introduction

The interest in extending the theory of formal languages led considerable research effort towards the two-dimensional languages where words can be considered as pictures. Several approaches have been proposed during the years; consequently, a general classification and a detailed comparison of the classes proposed turn to be necessary, though this work can be really hard. Lots of proposed formalisms are based on different approaches (automata-based or grammar-based are the most common ones) whose different nature results, almost always, in classes of languages which are incomparable each other.

In these recent years, much work is devoted to the study of formalisms which can be classified as the ones defining the regular languages class and showing interesting properties, i.e., closure properties, polynomial membership, power of expressiveness. The most accepted formalism as candidate for regular languages is the class *REC* introduced by Restivo and Giammaresi in [4]; it is defined by the family of Tiling Systems (*TS*) which can be considered as a generalization to two dimensions of the automata recognizing mono-dimensional words. The class is shown to be closed under all Boolean operations, row/column concatenations but not under complement and the membership problem results to be NP-complete. A *TS* language is given by projectioning symbols of a local language (defined by overlapping matching edges of square tiles of four characters) to symbols of the final language. The class of languages resulting from local composition of tiles is named as *LOC*.

Grammars can be considered as an alternative approach to define a language. Informally, two-dimensional grammars, so far proposed, can be classified into two distinct groups: *isometric* grammars transform a picture by means of rules which do not modify the dimensions of the area on which they are applied

and, so, of the whole picture. Conversely, *non-isometric* grammars consist of rules which can modify the dimensions of pictures; they transform a starting axiom by means of successive expansions of its sub-pictures, i.e., either terminals or non-terminals are replaced by a sub-pictures according to the rules defined. Recently, in [3],[2], it is proposed an isometric family of grammars, named as Tile Grammars (*TG*), based on the notion of picture tiling, which encompasses most of the existing grammar models and known formalisms. A restriction of the family, called Regional Tiling Grammars (*RTG*), is also introduced: it is more expressive than several existing formalisms, but incomparable with *REC*, yet it offers a polynomial-time parsing algorithm. Průša [7] gave a different definition of Context-Free two-dimensional languages from previous ones. The formalism consists of a non-isometric family of grammars which yields a set of languages that is incomparable with *REC*. In [1],[6], the authors show the proper inclusion of the languages generated by Průša grammars in the family of languages generated by *RTG*. More recently, Subramanian *et al.* proposed in [12],[11] a very simple non-isometric grammar formalism, called (*R*)*P2DCFG*, which is based on rewriting rules. Two sort of rules are considered which work separately both on rows and columns. Although the nature of productions limits the expressive power of the formalism, grammars can be endowed with a control (regular) language which defines legal sequences of rules to be used in generating the two-dimensional words. The resulting family of grammars is rather expressive since all symbols can be possibly used as non-terminals symbols.

In this work, some new closure results of (*R*)*P2DCFL*, preliminaries results about possible extensions of the class *P2DCFL* considering more expressive control languages, and a comparison among the above families of languages are shown.

2 Preliminaries

The following notation and definitions are mostly from [4].

Definition 1. Let Σ be a finite alphabet. A two-dimensional array of elements of Σ is a picture over Σ . The set of all pictures over Σ is Σ^{++} . A picture language is a subset of Σ^{++} . For $h, k \geq 1$, $\Sigma^{(h,k)}$ denotes the set of pictures of size (h, k) (where $|p| = (h, k)$, $|p|_{row} = h$, $|p|_{col} = k$). The symbol $\# \notin \Sigma$ is used when needed as a boundary symbol; \hat{p} refers to the bordered version of picture p , as shown in Fig. 2. A pixel is an element $p(i, j)$ of p . If all pixels are identical to $C \in \Sigma$ the picture is called C -homogeneous. The domain of a picture p is the set $dom(p) = \{1, 2, \dots, |p|_{row}\} \times \{1, 2, \dots, |p|_{col}\}$. Row and column concatenations are denoted \ominus and \oplus , respectively. $p \ominus q$ is defined iff p and q have the same number of columns; the resulting picture is the vertical juxtaposition of p over q . $p^{k\ominus}$ is the vertical juxtaposition of k copies of p ; $p^{+\ominus}$ is the corresponding closure. $\oplus, {}^{k\oplus}, {}^{+\oplus}$ are the column analogous. Let p be a picture over Σ . A subdomain of $dom(p)$ is a set d of the form $\{x, x+1, \dots, x'\} \times \{y, y+1, \dots, y'\}$ where $1 \leq x \leq x' \leq |p|_{row}$, $1 \leq y \leq y' \leq |p|_{col}$; also denoted as $(x, y; x', y')$. The set of subdomains of p is denoted $D(p)$. A subdomain is C -homogeneous (or homogeneous) when all pixels in the associated subpicture are

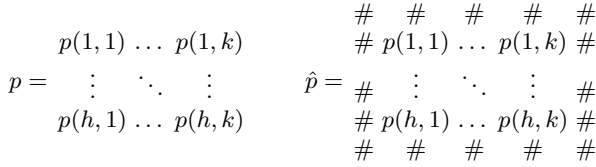


Fig. 1. A picture p and the corresponding bordered picture \hat{p}

identical to $C \in \Sigma$. Two subdomains $d_a = (i_a, j_a; k_a, l_a)$ and $d_b = (i_b, j_b; k_b, l_b)$ are horizontally adjacent (resp. vertically adjacent) iff $j_b = l_a + 1$, and $k_b \geq i_a, k_a \geq i_b$ (resp. $i_b = k_a + 1$, and $l_b \geq j_a, l_a \geq j_b$). We will call two subdomains adjacent, if they are either vertically or horizontally adjacent. A homogeneous partition of a picture p is any partition $\pi = \{d_1, d_2, \dots, d_n\}$ of $\text{dom}(p)$ into homogeneous subdomains d_1, d_2, \dots, d_n . The unit partition of p , written $\text{unit}(p)$, is the homogeneous partition of $\text{dom}(p)$ defined by single pixels. An homogeneous partition is called strong if adjacent subdomains have different labels.

If a picture p admits a strong homogeneous partition of $\text{dom}(p)$ into subdomains, then the partition is unique and will be denoted by $\Pi(p)$.

We now show the definitions of the formalisms we will consider in the paper. Tiling Systems define a notion of recognizability by means of projection of local properties. A picture over the alphabet Σ is a projection of an isometric picture over a local underlying alphabet Γ . Tiles can be considered as local “automata transitions”; and the process of “tiling” corresponds to a run of the automaton on the picture. Tiling Systems are defined as follows:

Definition 2. A tile is a square picture of size $(2, 2)$; let denote by $\llbracket p \rrbracket$ the set of all tiles contained in a picture p . If Σ is a finite alphabet, a (two-dimensional) language $L \subseteq \Sigma^{++}$ is local if there exists a finite set Θ of tiles over the alphabet $\Sigma \cup \{\#\}$ such that $L = \{p \in \Sigma^{++} \mid \llbracket \hat{p} \rrbracket \subseteq \Theta\}$. Let call such a language $\text{LOC}(\Theta)$. A tiling system (TS) is a 4-ple $T = (\Sigma, \Gamma, \Theta, \pi)$, where Σ and Γ are two finite alphabets, Θ is a finite set of tiles over the alphabet $\Gamma \cup \{\#\}$, and $\pi : \Gamma \rightarrow \Sigma$ is a projection.

Pure CF grammars which make use of only terminal symbols have been well investigated in the theory of string languages. Pure 2D context-free grammars [12], unlike Matrix grammars ([9], [10]), admit rewriting any row/column of pictures with no priority of columns and rows. Row/column sub-arrays of pictures are rewritten in parallel by equal length strings and by using only terminal symbols, as in a pure string grammar.

Definition 3. A Pure 2D Context-Free grammar (P2DCFG) is a 4-tuple $G = (\Sigma, P^c, P^r, S')$ where:

1. Σ is a finite set of symbols;
2. $P^c = \{c_i \mid 1 \leq i \leq m\}$, where a column rule c_i is a set of context-free rules of the form $a \rightarrow \alpha$, $a \in \Sigma$, $\alpha \in \Sigma^+$ s.t. for any two rules $a \rightarrow \alpha$, $b \rightarrow \beta$ in c_i , $|\alpha| = |\beta|$ where $|\alpha|$ denotes the length of α ;

3. $P^r = \{r_i \mid 1 \leq i \leq n\}$, where a row rule r_i is a set of context-free rules of the form $c \rightarrow \gamma^T$, $c \in \Sigma$, $\gamma \in \Sigma^+$ s.t. for any two rules $c \rightarrow \gamma^T$, $d \rightarrow \delta^T$ in r_i , $|\gamma| = |\delta|$;
4. $S' \subseteq \Sigma^{++}$ is a finite set of axioms.

Derivations are defined as follows: for any two arrays p_1, p_2 , we write $p_1 \Rightarrow p_2$ if p_2 is obtained from p_1 by either rewriting a column of p_1 by rules of some column table c_i in P^c or a row of p_1 by rules of some row table r_i in P^r . The reflexive transitive closure of \Rightarrow is denoted as $\overset{*}{\Rightarrow}$.

The picture array language $\mathcal{L}(G)$ generated by G is the set of pictures $\{p \mid S \overset{*}{\Rightarrow} p \in \Sigma^{++}$ for some $S \in S'\}$. The family of picture array languages generated by Pure 2D Context-free grammars is denoted by $P2DCFL$.

It is worth noticing that all pictures derived at each step by applying a rewriting rule from the set P^c or P^r are legal pictures. Since non-terminals are not admitted by the $P2DCFG$, each derivation consists of pictures in Σ^{**} .

To augment the expressive power of the $P2DCFG$ grammars, the sequence of rules to be used can be controlled by requiring it represents a word that belongs to a specific language. Generally, if the control language constitutes a regular language, the generative power of a grammar, generating a mono-dimensional language, may not increase; but this does not hold for two-dimensional grammars, as in the case of $P2DCFG$.

Definition 4. A Pure 2D Context-Free grammar with a regular control ($RP2DCFG$) is a tuple $G_r = (G, \Gamma, \mathcal{C})$ where:

1. G is a $P2DCFG$ grammar;
2. Γ is a set of labels of the rules of P^c and P^r ;
3. $\mathcal{C} \subseteq \Gamma^*$ is a regular language associated to the grammar.

Derivations are done as in G except that p is derived from S by means of a control word w , $S \Rightarrow_w p$, if $w \in \mathcal{C}$ and p is generated from S by applying the sequence of rule w . The picture language generated by G_r consists of all the pictures obtained from G with derivations controlled by \mathcal{C} .

The expressiveness of $RP2DCFG$ is greater than the one of $P2DCFG$. To demonstrate this fact we first observe that a $P2DCFG$ is a $RP2DCFG$ where $\mathcal{C} = \Gamma^*$. Moreover, the language of squares over the symbol a does not belong to $P2DCFL$ family but can be generated by a $RP2DCF$ grammar $(G, \{c, r\}, (cr)^*)$ where $G = (\{a\}, \{c\}, \{r\}, S)$ and

$$S \rightarrow a; \quad c : \{a \rightarrow aa\}, \quad r : \left\{ a \rightarrow \begin{matrix} a \\ a \end{matrix} \right\}. \tag{1}$$

The control language can belong to more expressive classes like context-free or context-sensitive languages. In [12], none of them are not considered but the generative power of $P2DCFG$ augments depending on the expressiveness of the family of the control language. In the next section, we consider the class

of $CF2DCFL$ generated by a $P2DCFG$ endowed with a context-free control language. It will be demonstrated that the family of $RP2DCFL$ is strictly contained in the one of $CFP2DCFL$.

For our purpose, in order to refine the given definition of this class of grammars, we consider $(R)P2DCFG$ whose alphabet is $\Sigma = \Sigma_T \cup \Sigma_C$ where Σ_T is the alphabet of final symbols defining the words and Σ_C is a set of auxiliary characters, *control symbols*, that are not intended to be symbols of the final pictures of the language but they are involved only in the process of derivation. Yet, control symbols can not be considered as proper non-terminal symbols since they have to be rewritten by means of derivations guided by the control language, so that no control symbol appear in the final picture. As shown in the Section 4, the use of an auxiliary alphabet of control symbols increases the expressiveness of the family of languages defined by $RP2DCFG$ as in [12].

The definition of Průša Grammar is taken from [8] and adapted as in [1], [6]. The formalism extends the generative power of CF Kolam grammars [10], [5], since it admits rules in which non-terminal symbols can be substituted with two-dimensional subpictures.

Definition 5. A 2D CF Průša grammar (PG) is a tuple (Σ, N, R, S) , where Σ is the finite set of terminal symbols, disjoint from the set N of nonterminal symbols; $S \in N$ is the start symbol; and $R \subseteq N \times (N \cup \Sigma)^{++}$ is the set of rules.

Let $G = (\Sigma, N, R, S)$ be a PG. We define a picture language $\mathcal{L}(G, A)$ over Σ for every $A \in N$. The definition is given by the following recursive descriptions:

1. if $A \rightarrow w$ is in R , and $w \in \Sigma^{++}$, then $w \in \mathcal{L}(G, A)$;
2. let $A \rightarrow w$ be a production in R , $w = (N \cup \Sigma)^{(m,n)}$, for some $m, n \geq 1$, and $p_{i,j}$, with $1 \leq i \leq m$, $1 \leq j \leq n$, be pictures such that:
 - (a) if $w(i, j) \in \Sigma$, then $p_{i,j} = w(i, j)$;
 - (b) if $w(i, j) \in N$, then $p_{i,j} \in \mathcal{L}(G, w(i, j))$;
 - (c) if $P_k = p_{k,1} \oplus p_{k,2} \oplus \dots \oplus p_{k,n}$, for any $1 \leq i \leq m$, $1 \leq j \leq n$, $|p_{i,j}|_{col} = |p_{i+1,j}|_{col}$, and $P = P_1 \oplus P_2 \oplus \dots \oplus P_m$; then $P \in \mathcal{L}(G, A)$.

The set $\mathcal{L}(G, A)$ contains exactly the pictures that can be obtained by applying a finite sequence of rules (i) and (ii). The language $\mathcal{L}(G)$ generated by grammar G is denoted as $\mathcal{L}(G, S)$.

Tile Grammars (TG) [3] perform an isometric derivation process for which homogeneous subpictures are replaced with isometric pictures of the local language defined by the right part of the rules. The derivation process starts from a S (axiom)-homogeneous picture and terminates when all non-terminals have been eliminated. A rule $A \rightarrow \omega$ defines an unbounded number of isometric pairs of pictures. The left part consists of all A -homogeneous pictures and the right part of all pictures of a local language over nonterminal symbols.

Definition 6. A tile grammar (TG) is a tuple (Σ, N, S, R) , where Σ is the terminal alphabet, N is a set of non-terminal symbols, $S \in N$ is the starting symbol, R is a set of rules. Let $A \in N$. There are two kinds of rules:

1. fixed size: $A \rightarrow t$, where $t \in \Sigma$;
2. variable size: $A \rightarrow \omega$, ω is a set of tiles over $N \cup \{\#\}$.

Consider a tile grammar $G = (\Sigma, N, S, R)$, let $p, p' \in (\Sigma \cup N)^{(h,k)}$ be pictures of identical size. Let $\pi = \{d_1, \dots, d_n\}$ be a homogeneous partition of $\text{dom}(p)$. We say that (p', π') derives in one step from (p, π) , written $(p, \pi) \Rightarrow_G (p', \pi')$ iff, for some $A \in N$, there exist in π an A -homogeneous subdomain $d_i = (x, y; x', y')$, called application area, and a rule $A \rightarrow \alpha \in R$ such that p' is obtained substituting $\text{spic}(p, d_i)$ in p with:

1. $\alpha \in \Sigma$, if $A \rightarrow \alpha$ is of type (1); i.e., $x = x'$ and $y = y'$;
2. $s \in \text{LOC}(\alpha)$, if $A \rightarrow \alpha$ is of type (2) and s admits a strong homogeneous partition $\Pi(s)$.

Moreover, $\pi' = (\pi \setminus \{d_i\}) \cup (\Pi(s) \oplus (x-1, y-1))$. We say that (p', π') derives from (p, π) in n steps, written $(p, \pi) \xrightarrow{n}_G (p', \pi')$, iff $p = p'$ and $\pi = \pi'$, when $n = 0$, or there are a picture p'' and a homogeneous partition π'' such that $(p, \pi) \xrightarrow{n-1}_G (p'', \pi'')$ and $(p'', \pi'') \Rightarrow_G (p', \pi')$. We use the abbreviation $(p, \pi) \xrightarrow{*}_G (p', \pi')$ for a derivation with a finite number of steps.

At each step of the derivation, an A -homogeneous subpicture is replaced with an isometric picture of the local language defined by the right part α of a rule $A \rightarrow \alpha$, where α admits a strong homogeneous partition. The process terminates when all nonterminals have been eliminated from the current picture.

Definition 7. The picture language defined by a grammar G (written $\mathcal{L}(G)$) is the set of $p \in \Sigma^{++}$ such that $(S^{|p|}, \{\text{dom}(p)\}) \xrightarrow{*}_G (p, \text{unit}(p))$. For short we also write $S \xrightarrow{*}_G p$.

Regional Tile Grammars (RTG) [1], [6] are specialization of Tile Grammars.

Definition 8. A homogeneous partition is regional (HR) iff distinct (not necessarily adjacent) subdomains have distinct labels. A picture p is regional if it admits a HR partition. A language is regional if all its pictures are so. A regional tile grammar (RTG) is a tile grammar (see Definition 6), in which every variable size rule $A \rightarrow \omega$ is such that $\text{LOC}(\omega)$ is a regional language.

3 Closure Properties and Hierarchy

In this section some closure properties of the class of $(R)P2DCFL$ are investigated. The $P2DCFL$ family was shown not to be closed under union in [12]. This is not the case of $RP2DCFL$ as shown by the following:

Proposition 1. Let $G_r^1 = (G_1, \Gamma_1, C_1)$ and $G_r^2 = (G_2, \Gamma_2, C_2)$ be two $RP2DCFG$. Then, the language $\mathcal{L}(G_r^1) \cup \mathcal{L}(G_r^2)$ is $RP2DCFL$.

Proof. Let $G_1 = (\Sigma_1, P_1^c, P_1^r, S_1)$ and $G_2 = (\Sigma_2, P_2^c, P_2^r, S_2)$ be two $RP2DCFG$ grammars. It is possible to define the grammar for $\mathcal{L}(G_r^1) \cup \mathcal{L}(G_r^2)$ by renaming the rules c_i/r_i of G_j to c_i^j/r_i^j where $j \in \{1, 2\}$. So, each rule of G_1 and G_2 is indexed by the respective index naming the grammar itself. Let $\bar{\Gamma}_1$ and $\bar{\Gamma}_2$ the

two renamed alphabets naming the row/column the productions, \bar{P}_i^c and \bar{P}_i^r , where $i \in \{1, 2\}$, be the set of productions and \bar{C}_1 and \bar{C}_2 be the two control languages over the renamed control alphabets, derived from the previous ones, C_1 and C_2 , by replacing each c_i/r_i (of the grammar G^j) with the respective c_i^j/r_i^j . The final grammar is the tuple $G_r^{1\cup 2} = (G^{1\cup 2}, \bar{I}_1 \cup \bar{I}_2, \bar{C}_1 \cup \bar{C}_2)$ where $G^{1\cup 2} = (\Sigma_1 \cup \Sigma_2, \bar{P}_1^c \cup \bar{P}_1^r, \bar{P}_1^r \cup \bar{P}_1^c, S_1 \cup S_2)$. \square

The family of *P2DCFL* was shown not to be closed under row/column concatenation in [12]; this holds also for the family of *RP2DCFL* as shown here.

Proposition 2. *The family of RP2DCFL is not closed under row/column concatenation.*

Proof (hint). Let us consider two *RP2DCFG* grammars G_r^1 and G_r^2 defined in the following way over the alphabet $\Sigma = \{a, b\}$:

1. $G_r^1 = (G_1, \Gamma_1, \mathcal{C}_1)$, where
 - $G_1 = (\Sigma, \{c\}, \{r\}, S)$;
 - $S \rightarrow a$;
 - $\Gamma_1 = \{c, r\}$;
 - $c : \{a \rightarrow ab, b \rightarrow ba\}, r : \left\{ a \rightarrow \begin{smallmatrix} a \\ b \end{smallmatrix}, b \rightarrow \begin{smallmatrix} b \\ a \end{smallmatrix} \right\}$;
 - $\mathcal{C}_1 = (cr)^*$;
2. $G_r^2 = (G_2, \Gamma_2, \mathcal{C}_2)$, where
 - $G_2 = (\Sigma, \{c'\}, \{r\}, S)$;
 - $\Gamma_2 = \{c', r\}$;
 - $c' : \{a \rightarrow ab, b \rightarrow ab\}$;
 - $\mathcal{C}_2 = (c'r)^*$.

The two grammars produce two different languages of squares of a, b . Since the row production r does not alter the rewritten symbols a, b (it, indeed, rewrites a, b into a couple $a/b, b/a$ where the first position is again a, b) then the first row of the two squares is defined by two different string languages: $L_1 = a(a^*b^+)^+$ and $L_2 = a(b^+a^*)^+$. It is possible to show the grammar defining the column concatenation of the previous languages needs a CF control though a generic language of rectangles $(n, 2n)$ of a, b can be defined by a *RP2DCFG*. The production $b \rightarrow ba$ and $b \rightarrow ab$ are in conflict; they can be only distinguished by renaming one of the two left part, e.g., $b' \rightarrow ab'$ for G_2 . But each occurrence of b in the production set of G_2 must be renamed, in turn, to avoid possible new conflicts. The grammar of the language equivalent to the column concatenation is given by $G_{12} = (\Sigma_{12}, P_{12}^c, P_{12}^r, S_{12})$ where $\Sigma_{12} = \{a, b, a', b'\}$, $P_{12}^c = \{c_1, c_2\}$, $P_{12}^r = \{r_{12}\}$, $S_{12} \rightarrow aa'$ and $r_{12} : \left\{ a \rightarrow \begin{smallmatrix} a \\ b \end{smallmatrix}, b \rightarrow \begin{smallmatrix} b \\ a \end{smallmatrix}, a' \rightarrow \begin{smallmatrix} a' \\ b' \end{smallmatrix}, b' \rightarrow \begin{smallmatrix} b' \\ a' \end{smallmatrix} \right\}$, $c_1 = c$, $c_2 = c'$ and each occurrence of a, b in c_2 is primed, endowed with a regular control $\mathcal{C}_{12} = (c_1c_2r_{12})^*$. Each derivation yields a picture composed by two squares: the left one over $\{a, b\}$ and the right one over $\{a', b'\}$. Each occurrence of primed characters have to be changed into the corresponding non-primed characters of the final alphabet $\{a, b\}$ by means of the rule: $f : \{a' \rightarrow a, b' \rightarrow b\}$. Since the

dimension of the squares is arbitrarily defined, and the rule f works only on one row/column at a time, the control language became: $(c_1c_2r_{12})^n f^n$ which is not regular. \square

Corollary 1. *The family of RP2DCFL language is strictly included in the family CFP2DCFL.*

Proposition 3. *The family of P2DCFL is not closed under intersection.*

Proof. It is possible to provide a counterexample which shows the intersection of two languages in P2DCFL yields a language in RP2DCFL. Let $L_{rect(a)} = \{p \in \{a\}^{**}\}$, $L_{square(a)} = \{p \in \{a\}^{**} \mid |p|_{row} = |p|_{col}\}$ be the languages of rectangles and squares, respectively, of a . The grammar in (II), shown above, defines $L_{square(a)}$; rectangles are given by the same grammar without the control language. Let L be the language given by the union of the following three classes of languages: (i) squares of a bordered by a right column of b , a row of c on the bottom and a d in the right-bottom corner; (ii) rectangle of a bordered by a right column of b , a row of c on the bottom and a e in the right-bottom corner and (iii) the class of squares of a . The language L is defined by the grammar $G = (\Sigma, P^c, P^r, S)$ where $\Sigma = \{a, b, c, d\}$, $P^c = \{c, f\}$, $P^r = \{r\}$,

$$c : \{b \rightarrow ab \mid a, d \rightarrow ce \mid a\}, r : \left\{ c \rightarrow \begin{matrix} c \\ c \end{matrix} \mid a, e \rightarrow \begin{matrix} b \\ d \end{matrix} \right\},$$

$$f : \{b \rightarrow a, d \rightarrow a, d \rightarrow a\}, \quad S \rightarrow \begin{matrix} ab \\ cd \end{matrix}$$

The rule f is, possibly, used to derive squares of a . It follows that $L_{rect(a)} \cap L$ is equal to $L_{square(a)}$, which requires a control regular language to be defined. \square

4 Comparisons

In this paragraph, we give results concerning the relations of the class RP2DCFL with respect to LOC, $\mathcal{L}(PG)$ and $\mathcal{L}(RTG)$.

Let us consider the language of words containing an arbitrary number of diagonals of 1 which consist of at least two characters (no single 1 are admitted at the corners) and that are separated by at least one diagonal of 0. Let this language be named as L_{diag} . It can be shown to be LOC by showing the tile-set of the tiling system. The language does not belong to RP2DCFL: its restriction considering only two diagonals, $L_{diag(2)}$ is not in RP2DCFL. The idea of the proof is based on the fact that a finite number of terminals can only generate pictures of two diagonals of finite length (with finite reciprocal distance). To correctly put a symbol 1 of a diagonal by means of row/column rules, control terminals are used to define the position of the 1 in the picture, by starting the construction of the word from the reference boundary of the picture. Since the set of characters is finite, the set of pictures which can be derived is finite as well. Before the main theorem, some lemmas are here shown. Let $L_d = \{p \in \{0, 1\}^{++} \mid |p|_{row} = |p|_{col}, p(i, j) = 1 \text{ for } i = j, p(i, j) = 0 \text{ for } i \neq j\}$ be the language of main diagonals of 1 in a field of 0.

Lemma 1. *The language L_d can be defined by using at least one control character and a regular control language.*

Proof. Let p be a (square) picture of the language, with $|p|_{row} = |p|_{col} = n$. Let suppose to enlarge it by adding one row and one column, by means of one row (column) production followed by one column (row) production. Since the vertical/horizontal expansion of a character 1 is always 0 and the alphabet is $\{0, 1\}$, two characters 0 laying on a row (column) should yield two different expansions depending on their position with respect to the 1 on the same row (column). In Fig. 2 the first column derivation is depicted. Only the 0 in the right-bottom corner is in charge of placing the 1 below. Since the alphabet does not contain control characters, the vertical expansion can yield pictures which do not belong to the language. The same reasoning holds when considering a column construction. By using a control character \bullet and a regular control language which constraints a strict alternation of row and column rule, it is possible to define correct derivations:

$$\begin{array}{ccc}
 1\ 0\ 0 & 1\ 0\ 0\ 0 & 1\ 0\ 0\ 0 \\
 0\ 1\ 0 & \rightarrow 0\ 1\ 0\ 0 & \rightarrow 0\ 1\ 0\ 0 \\
 0\ 0\ 1 & 0\ 0\ 1\ \bullet & 0\ 0\ 1\ 0 \\
 & & 0\ 0\ 0\ 1
 \end{array} \quad \square$$

The following lemma shows that, by using a finite alphabet, a picture of two distinct diagonals can not be constructed from the origin points, without changing the relative position between them.

Lemma 2. *Let us consider $p \in L_{diag(2)}$ such that $p(|p|_{row}, 1) = 1$, $p(1, |p|_{col}) = 1$ are the starting points of the two diagonals; then, it is not possible to prolong the two diagonals unless the origins are moved in $p(|p|_{row} + \delta_r, 1)$, $p(1, |p|_{col} + \delta_c)$ where $\delta_r, \delta_c \geq 1$.*

Proof. From Lemma 1 it is known that two distinct diagonals can not be constructed by means of a binary alphabet; the construction requires at least one control character more. However, this construction moves the position of the origin points and makes the distance between them grow when the dimension of the picture grows. This fact follows, immediately, from the row/column generation of P2DCFG grammars, due to the following rules:

$$c : \{1 \rightarrow 1\ \bullet\}, \quad r : \left\{ 1 \rightarrow \begin{array}{l} 1 \\ 0 \end{array}, \bullet \rightarrow \begin{array}{l} 0 \\ 1 \end{array} \right\};$$

$$\begin{array}{ccc}
 1\ 0\ 0 & 1\ 0\ 0\ 0 \\
 0\ 1\ 0 & \rightarrow 0\ 1\ 0\ 0 \\
 0\ 0\ 1 & 0\ 0\ 1\ 0
 \end{array}$$

Fig. 2. Column expansion

or, alternatively

$$c : \{1 \rightarrow 1\ 0, \bullet \rightarrow 0\ 1\}, \quad r : \left\{ 1 \rightarrow \begin{matrix} 1 \\ \bullet \end{matrix} \right\} \quad \square$$

Corollary 2. *A set of control characters is required to prolong diagonals.*

From the last two lemmas, it is possible to prove the following:

Proposition 4. *The language $L_{diag(2)}$ is not in $RP2DCFL$.*

Proof. Let G be a $RP2DCFG$, $\mathcal{L}(G) = L_{diag(2)}$ and $\Sigma = \{0, 1\} \cup C$ where C is a set of control characters. If the cardinality of C is finite then the length of diagonals is finite. Let suppose p be a sub-picture in the considered language constructed by means of rules acting over characters in the set C . Since the cardinality of C is finite and since no control character can be derived yet from the picture, because of the ambiguity of the derivation as shown in Lemma 1, the set of words with two diagonals is finite. If it was not so, it could be possible to derive arbitrary diagonals by means of a finite alphabet without moving their origins. But this is not possible by construction, from the previous lemmas. Moreover, no row/column rule can be used to modify a picture. If they are applied outside the gray region, see Fig. 3, the origin of diagonals will be moved since any production will add at least either one row or one column in the area between the origins and all the characters of p are only $\{0, 1\}$. Moreover, any rule which makes use of control symbols, and which is applied on row/column of the gray region, will lead to picture not belonging in the language: one of the two diagonals is interrupted since a row/column is added between two consecutive symbols 1. This means that the rules which makes use of control symbols can not be used to derive pictures of diagonals before the gray area is obtained and also when it is actually built. Thus, the diagonals have to be derived by means of control characters and, then, are of finite length due to Lemma 2. However, by augmenting arbitrarily the distance of the origin points of diagonals, the area on which the set of rules which uses control symbols are not admitted will grow and, then, the length of diagonals as well. Since the set of control characters is finite, then the set of pictures resulting from the grammar is finite. \square

From here on, other families of languages are considered. The language of pictures with one diagonal of 1 of depth 2 can be shown to belong to LOC , $RP2DCFL$, $\mathcal{L}(RTG)$ and $\mathcal{L}(PG)$. A language of diagonals similar to the one presented above that can be used to further characterize LOC is the language of two minor diagonals of 1 beside the main one of 0, in a field of 0's, defined as $L = \{p \in \{0, 1\}^{++} \mid |p|_{row} = |p|_{col}, p(i, j) = 1 \text{ for } i - 1 = j \text{ and } i = j - 1; p(i, j) = 0 \text{ else}\}$. It goes beyond the expressive power of LOC since the number of diagonals of a picture of arbitrary size can not be controlled by a local definition. The required tile set necessary to generate all the pictures of the language is the same as the one of the language L_{diag} . To correctly define the language, a TS should be used. The main diagonals should be locally denoted

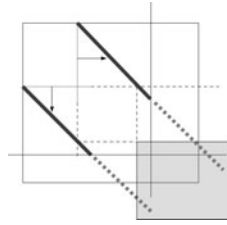


Fig. 3. Regions of a two diagonals picture

by a special symbol different from 0; and, by means of a projection, it has to be substituted by 0. The language is a *RP2DCFL*, also definable by a *PG* and, consequently, by a *RTG*.

Although the *RTG* class seems to be sufficiently expressive inasmuch as it contains lots of two-dimensional languages like *PG*, we show a language of pictures of symmetrical squares which can not be generated by any regional grammar. Consequently, the language does not belong to the family of languages resulting from *PG*. However, it can be generated by a *RP2DCFG*. A legal picture of this language is here depicted.

$$\begin{array}{c}
 a\ b\ b\ a \mid a\ b\ b\ a \\
 b\ b\ b\ a \mid a\ b\ b\ b \\
 b\ b\ b\ a \mid a\ b\ b\ b \\
 a\ a\ a\ a \mid a\ a\ a\ a
 \end{array}$$

Note that each square is composed by nested “L” shaped strings of the same character, over the alphabet $\Sigma = \{a, b\}$. We denote the language as L_{\perp} .

Proposition 5. *The language L_{\perp} is not in $\mathcal{L}(RTG)$.*

Proof. Let p be a picture of the language L_{\perp} . Let n be the length of the side of the two squares composing the picture, such that $|p|_{row} = |p|_{col} = n$. Then, the number of different pictures is 2^n , since the picture is symmetric. In order to define two square regions, the first production of the grammar should define them in the first derivation: since the partition is strong, the two sub-pictures generated from A and B are regionally defined, i.e., the derivations of the one can not affect the one of the other. So, the number of pictures “L” shaped which can be derived is, in general, 2^{2n} . Moreover, let us consider a rectangle area consisting of two areas of the same dimension, on the left and on the right of the vertical axis. If they belong to different regions, i.e., they are the derivation of two different non-terminals of a strong partition, there does not exist a method to make them symmetric because they are regionally defined. Otherwise, they belong to the same homogeneous partition and, then, there exists at least one derivation making them symmetric. But, by extending the partition to cover the whole picture, it is easy to see the two square regions are no longer correctly definable and, in general, the left and the right part can yield different sub-pictures. \square

Finally, the last language we consider does not belong to *RP2DCFL* but it can be generated by a *PG* and, a fortiori, by a *RTG*. The language is the set of

square pictures of size $(2^n, 2^n)$, where n is an arbitrary natural, i.e., $L_{(2^n, 2^n)} = \{p \in \{a\}^{**} \mid |p|_{row} = |p|_{col} = 2^h, h \geq 0\}$.

Proposition 6. *The language $L_{(2^n, 2^n)}$ is not in $RP2DCFL$.*

Proof. Let p_{2^i} be the square of size $(2^i, 2^i)$, w_{2^i} the string word representing its side and let α_{2^i} be the control word generating p_{2^i} . Obviously $w_{2^i} = aa \dots a^{2^{i-1}}$. Let $L_S = \{w \in a^+ \mid w = w_{2^i}\}$ be the language of the sides. Let α_{2^i} be the control word generating the picture of side w_{2^i} . Let us consider a subset of $L_{(2^n, 2^n)}$ such that each $|\alpha_{2^k}| \leq n_q$, where n_q is the number of states of the minimal automaton recognizing the control language. By geometric construction, the pictures of $L_{(2^n, 2^n)}$ such that $w_{2^{k+1}} \in L_S, \dots, w_{2^{k+h}} \in L_S$ are recursively derived by row/column concatenation of pictures whose sides are $w_1 \dots w_{2^k}$. A square of sides $w_{2^{k+1}}$ consists of four isometric squares of sides w_{2^k} such that $w_{2^{k+1}} = w_{2^k} w_{2^k}$; each w_{2^k} is, in turn, a row/column concatenation of pictures of sides $w_1 \dots w_{2^{k-1}}$. In general, a square of sides $w_{2^{k+h}}$ consists of h^2 isometric squares such that $w_{2^{k+h}} = (w_{2^k})^h (w_{2^k})^h$. Then, to generate a generic picture of side $w_{2^{k+h}} = (w_{2^k})^h (w_{2^k})^h$ it is required a control word $(\alpha_{2^k})^n (\alpha_{2^k})^n$. Since the control language is regular, from the Pumping lemma, this can be actually done only for a finite interval of values of n . On the other hand, the Parikh map of the control language is a semilinear set. Then, the length of sides of pictures which result from the application of the rules of the grammar, led by a regular control, is a linear combination $p_1|w_1| + \dots + p_k|w_{2^k}|$ with $p_i \geq 0$ for $1 \leq i \leq 2k$, of the length of sides of pictures defined by control words $|\alpha_{2^i}| \leq n_q$. Then, the resulting set of pictures exceeds the language since any exponential function can not be written as linear combination of terms. \square

In order to distinguish two classes of languages, it could be useful to analyze the derivation process of pictures. In the case of $RP2DCFG$, the number of control symbols involved in each picture results to be bounded. Let $\gamma : \Sigma^{**} \rightarrow \mathbb{N}$ be the function which counts the number of control symbols involved in a picture. Then, the following proposition holds:

Proposition 7. *Let L be a language in $RP2DCFL$. If $p \in L$ derived from S , $S \xrightarrow{*} p_i \xrightarrow{*} p$ then there exists $k \in \mathbb{N}$ such that $\gamma(p_i) < k$.*

Proof. Let A be the subpicture generated by the application of the left part α_i of rules of a row/column production. By definition, A is finite and consists of symbols of Σ_T and symbols of Σ_C . Each symbol of Σ_C in A must be eventually substituted by a symbol in Σ_T , since the final picture $p \in \Sigma^{**}$. Let n_q be the number of states of the minimal automaton recognizing the (regular) control language. Consequently, since the control language is regular, the number of rules involved in rewriting of all the control symbols, during the derivation of a picture p , is finite. Let $S \Rightarrow p_1 \xrightarrow{*} p_i \xrightarrow{*} p$ be the derivation of p from the axiom S ; then, the number of substitutable control symbols at each step is bounded by n_q , i.e., $\gamma(p_i) \leq n_q$. \square

5 Conclusions

In this work, some two-dimensional formalisms were compared each other. The goal the authors intended to do, was to refine the knowledge concerning their relationships. Future works concern the study of the hierarchy of $P2DCFG$ endowed with a CF or a CS control language. It is still an open problem, also for unary alphabet, if to the one-dimensional hierarchy of Chomsky may correspond a $P2DCFL$ hierarchy based on the expressiveness of the control language. The membership algorithm recognizing pictures of a $(R)P2DCFL$ should be studied and the intersection of $\mathcal{L}(RTG)$ and $RP2DCFL$ as well.

Acknowledgement. We would like to thank Stefano Crespi Reghizzi and Matteo Pradella for the support and suggestions given us during the development of this work.

References

1. Cherubini, A., Crespi-Reghizzi, S., Pradella, M.: Regional languages and tiling: A unifying approach to picture grammars. In: Ochmański, E., Tyszkiewicz, J. (eds.) MFCS 2008. LNCS, vol. 5162, pp. 253–264. Springer, Heidelberg (2008)
2. Cherubini, A., Crespi-Reghizzi, S., Pradella, M., San Pietro, P.: Picture languages: Tiling systems versus tile rewriting grammars. *Theor. Comput. Sci.* 356(1-2), 90–103 (2006)
3. Crespi-Reghizzi, S., Pradella, M.: Tile rewriting grammars and picture languages. *Theor. Comput. Sci.* 340(1), 257–272 (2005)
4. Giammarresi, D., Restivo, A.: Ambiguity and complementation in recognizable two-dimensional languages. In: IFIP TCS, pp. 5–20 (2008)
5. Matz, O.: Regular expressions and context-free grammars for picture languages. In: Reischuk, R., Morvan, M. (eds.) STACS 1997. LNCS, vol. 1200, pp. 283–294. Springer, Heidelberg (1997)
6. Pradella, M., Cherubini, A., Crespi-Reghizzi, S.: A unifying approach to picture grammars. CoRR, arXiv:0910.2829 (2009)
7. Průša, D., Hlavác, V.: 2D context-free grammars: Mathematical formulae recognition. In: *Stringology*, pp. 77–89 (2006)
8. Průša, D.: Two-dimensional Languages. PhD thesis, Charles University, Faculty of Mathematics and Physics, Czech Republic (2004)
9. Siromoney, G., Siromoney, R., Krithivasan, K.: Abstract families of matrices and picture languages. *Computer Graphics and Image Processing* 1, 284–307 (1972)
10. Siromoney, G., Siromoney, R., Krithivasan, K.: Picture languages with array rewriting rules. *Information and Control* 23(5), 447–470 (1973)
11. Subramanian, K.G., Ali, R.M., Geethalakshmi, M., Nagar, A.K.: Pure 2D picture grammars and languages. *Discrete Applied Mathematics* 157(16), 3401–3411 (2009)
12. Subramanian, K.G., Nagar, A.K., Geethalakshmi, M.: Pure 2D picture grammars (P2DPG) and P2DPG with regular control. In: Brimkov, V.E., Barneva, R.P., Hauptman, H.A. (eds.) IWCIA 2008. LNCS, vol. 4958, pp. 330–341. Springer, Heidelberg (2008)

Petri Net Generating Hexagonal Arrays

D. Lalitha¹, K. Rangarajan², and D.G. Thomas³

¹ Department of Mathematics, Sathyabama University
Chennai - 600 119, India
lalkrish_24@yahoo.co.in

² Department of Mathematics, Barath University, Chennai - 600 073, India
kr2210@hotmail.com

³ Department of Mathematics, Madras Christian College
Tambaram, Chennai - 600 059, India
dgthomasmcc@yahoo.com

Abstract. A new model to generate hexagonal arrays using Petri net structure has been defined. The catenation of an arrowhead to a b -hexagon results in a similar b -hexagon. This concept has been used in Hexagonal Array Token Petri Net Structure (HATPNS). A variation in the position of catenation has been introduced in Adjunct Hexagonal Array Token Petri Net Structure (AHATPNS). Comparisons with other hexagonal array models have been made.

Keywords: Hexagonal picture languages, Petri nets, arrowhead catenations, adjunct rules, array tokens.

1 Introduction

Hexagonal arrays and hexagonal patterns are found in the literature on picture processing and scene analysis. Image generation can be done in many ways in formal languages. Several grammars were introduced in the literature to generate various classes of hexagonal picture languages. The class of Hexagonal Kolam Array Languages (HKAL) was introduced by Siromoneys [7]. The class of Hexagonal Array Languages (HAL) was introduced by Subramanian [8]. The class of local and recognizable hexagonal picture languages were introduced by Dersanambika et al. [1].

On the other hand, a Petri net is an abstract formal model of information flow [3]. Petri nets have been used for analysing systems that are concurrent, asynchronous, distributed, parallel, non deterministic and / or stochastic. Tokens are used in Petri nets to simulate dynamic and concurrent activities of the system. A language can be associated with the execution of a Petri net. By defining a labeling function for transitions over an alphabet, the set of all firing sequences, starting from a specific initial marking leading to a finite set of terminal markings, generates a language over the alphabet.

Petri net model to generate rectangular arrays has been introduced in [5]. Motivated by this concept we have introduced a Petri net model to generate

hexagonal picture languages. In this model hexagonal arrays over a given alphabet are used as tokens in the places of the net. Labeling of transitions are defined as arrowhead catenation rules. Firing a sequence of transitions starting from a specific initial marking leading to a finite set of terminal markings would catenate the arrowheads to the initial array and move the array to the final set of places. The collection of such arrays is defined as the language generated by the Petri net structure. We call the resulting model as Hexagonal Array Token Petri Net Structure (HATPNS). The generative capacity of HATPNS is compared with HKAG [7] and HAG [8]. Various positions of adjunction have been defined to join the arrowhead into the hexagonal array of the input place. With these adjunction rules as labels the firing sequence would move the input array to the final place after joining the various arrowheads. This model is called Adjunct Hexagonal Array Token Petri Net Structure (AHATPNS). The generative capacity of this model is compared with controlled table 0L/1L hexagonal array grammars, extended 2D hexagonal context-free grammar and HATPNS.

One application for such a generation is in biomedical image processing [6]. A programmable cellular automaton is used for processing biomedical images. The cellular register is arranged in a hexagonal pattern produced by alternatively switching the shift register stages selected from line to line. Other applications are in the field of tiling patterns and generation of kolam patterns.

The paper is organized as follows: Section 2 recalls the notions of hexagonal arrays and arrowheads. Section 3 introduces a new model HATPNS. Section 4 compares the generative powers of HATPNS with other existing models. Section 5 introduces another model AHATPNS and compares it with HATPNS.

2 Hexagonal Arrays and Arrowheads

Let Σ be a finite non empty set of symbols. The set of all hexagonal arrays made up of elements of Σ is denoted by Σ^{**H} . The size of any hexagonal array is defined by its parameters. For a hexagon the parameters are $|H|_{LU}$, $|H|_{RU}$, $|H|_R$, $|H|_{RL}$, $|H|_{LL}$ and $|H|_L$ where LU stands for left upper; RU , right upper; R , right; RL , right lower; LL , left lower and L , lower. A hexagon of Σ^{**H} is shown in Fig. 1(a) where $\Sigma = \{a\}$ and the sides of the hexagon are shown in Fig. 1(b).

For any hexagon three types of catenation with six arrowheads are possible.

For a hexagon H an arrowhead of type $A_1(A_2)$ can be catenated in the direction \downarrow (\uparrow) if and only if $|H|_{LL} = |A_1|_{LU}$, $|H|_{RL} = |A_1|_{RU}$ ($|H|_{LU} = |A_2|_{LL}$, $|H|_{RU} = |A_2|_{RL}$). Similarly the other two catenations are possible subject to the corresponding conditions [2,7]. The arrowheads for the hexagon in Fig. 1(a) are shown in Fig. 2. An arrowhead is written in the form $\{\dots \langle v \rangle \dots\}$ where $\langle v \rangle$ denotes the vertex and the arrowhead is written in the clockwise direction.

Hexagon has 6 sides namely 1, 2, 3, 4, 5, 6 as shown in Fig. 1(a). Then A_1 will be catenated with sides 4, 5; A_2 with sides 1, 2; A_3 with sides 2, 3; A_4 with sides 5, 6; A_5 with sides 6, 1 and A_6 with sides 3, 4.

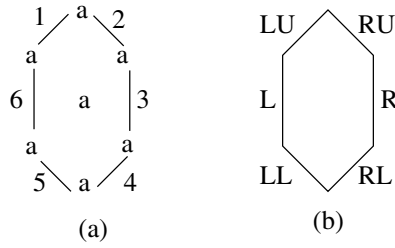


Fig. 1.

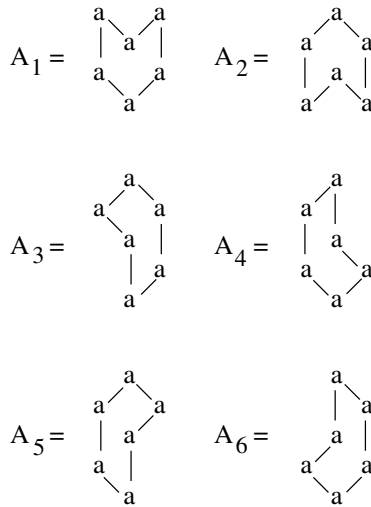


Fig. 2. Arrowheads for the hexagon in Fig. 1.

3 Hexagonal Array Token Petri Net Structure

Definition 1. A Petri net structure is a four tuple $C = (P, T, I, O)$ where $P = \{p_1, p_2, \dots, p_n\}$ is a finite set of places $n \geq 0$, $T = \{t_1, t_2, \dots, t_m\}$ is a finite set of transitions $m \geq 0$, $P \cap T = \phi$. $I : T \rightarrow P^\infty$ is the input function from transition to the bags of places and $O : T \rightarrow P^\infty$ is the output function from the transition to the bags of places.

Definition 2. A Petri net marking is an assignment of tokens to the places of a Petri net. The tokens are used to define the execution of the Petri net. The number and position of tokens may change during the execution of the Petri net.

Definition 3. If the tokens are arrays over a given alphabet Σ then the Petri net is an array token Petri net [5].

3.1 Firing Rules

A transition without any label will fire only if all the input places have the same hexagonal array as a token. Then on firing the transition arrays from all the input places are removed and put in all its output places. If all the input places have different arrays then the transition without label cannot fire. If the input places have different arrays then the label of the transition has to specify an input place. When the transition fires the arrays in the input places are removed and the array in the place specified in the label is put in all the output places.

3.2 Arrowhead Catenation Rules as Labels

Let a transition t have $H \otimes A$ as a label where \otimes is any one of the six directions $(\nearrow, \rightarrow, \searrow, \swarrow, \leftarrow, \nwarrow)$, H is the hexagonal array in all the input places and A a predefined arrowhead. Then firing the transition will catenate A with H in the specified direction and put in all the output places subject to the condition of catenation. If the condition for catenation is not satisfied then the transition cannot fire.

Example 1.

$$\text{Let } H = \begin{matrix} & a & \\ a & a & \\ a & a & \\ & a & \end{matrix}, \quad A_6 = \begin{matrix} & b & \\ & b & \\ & b & \end{matrix}, \quad H_1 = \begin{matrix} & a & \\ a & a & \\ a & a & \\ & a & \\ & b & \end{matrix}$$

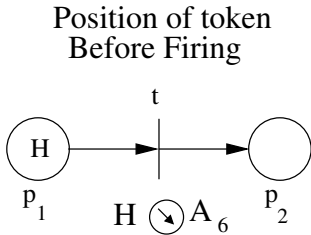


Fig. 3.

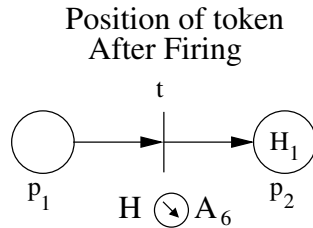


Fig. 4.

Definition 4. If $C = (P, T, I, O)$ is a Petri net structure with hexagonal arrays of Σ^{**H} as initial markings $M_0 : P \rightarrow \Sigma^{**H}$, labels of transitions being arrowhead catenation rules and a finite set of final places $F \subseteq P$, then the Petri net structure is defined as Hexagonal Array Token Petri Net Structure (HATPNS).

Definition 5. If C is a HATPNS then the language generated by the Petri net C is defined as

$$L(C) = \{H \in \Sigma^{**H} / H \text{ is in } p \text{ for some } p \in F\}$$

With hexagonal arrays of Σ in some places as initial marking all possible sequences of transitions are fired. The set of all arrays collected in the final places F is called the language generated by C .

Example 2. Let $\Sigma = \{X, \bullet\}$;

$$\begin{aligned}
 B_1 &= \begin{pmatrix} \bullet \\ \bullet \\ \bullet \end{pmatrix} \begin{matrix} |H|_{LU} - 1 \\ \\ \end{matrix} \begin{pmatrix} X \\ X \\ X \end{pmatrix} \begin{matrix} X & X & X \\ \bullet & \bullet & X \\ \bullet & \bullet & X \end{matrix}; \\
 B_2 &= \begin{matrix} X & X & X \\ X & \bullet & \bullet \\ X & \bullet & \bullet \end{matrix} \begin{pmatrix} X \\ X \\ X \end{pmatrix} \begin{pmatrix} \bullet \\ \bullet \\ \bullet \end{pmatrix} \begin{matrix} |H|_R - 2 \\ \\ \end{matrix} \begin{matrix} X \\ X \\ X \end{matrix}; \\
 B_3 &= \begin{matrix} X & X & X \\ X & \bullet & \bullet \\ X & \bullet & \bullet \end{matrix} \begin{pmatrix} X \\ X \\ X \end{pmatrix} \begin{pmatrix} X \\ \bullet \\ \bullet \end{pmatrix} \begin{matrix} |H|_R - 2 \\ \\ \end{matrix} \begin{matrix} X \\ X \\ X \end{matrix} \text{ and} \\
 F &= \{p_5\}.
 \end{aligned}$$

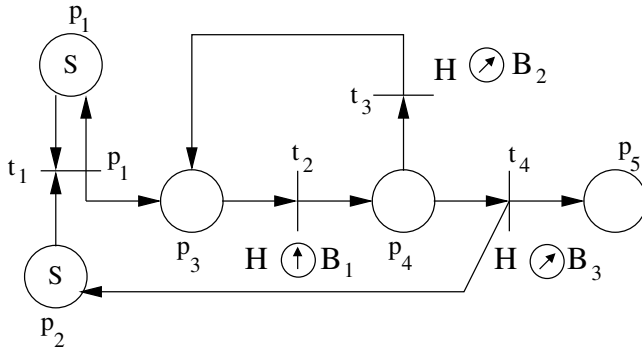
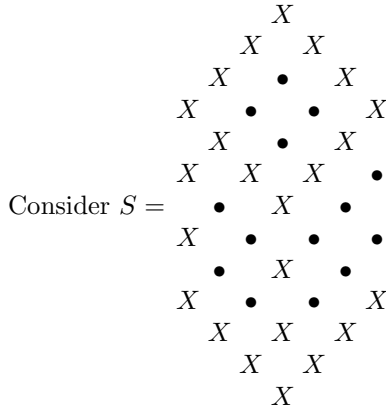
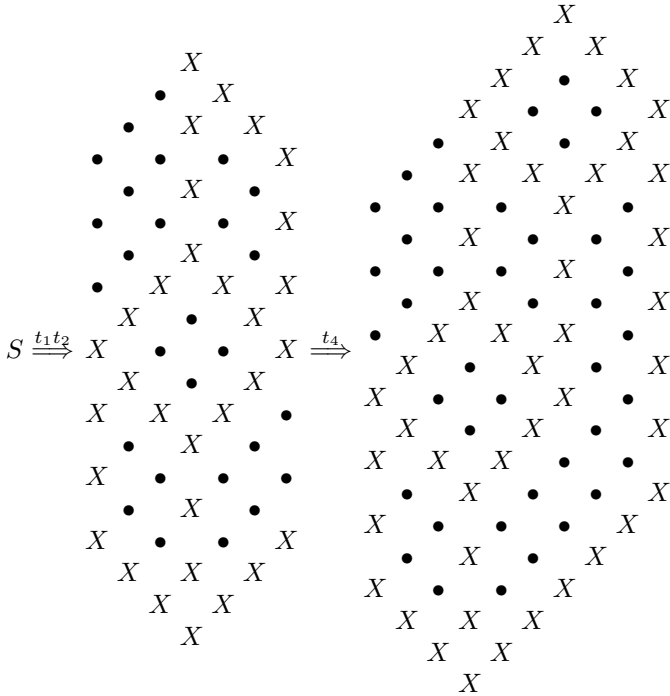


Fig. 5. HATPNS

When t_1 fires the token from place p_1 moves to p_3 . Derivation of the first array of the language is shown below:



The firing sequence $t_1(t_2t_3)^{n-1}t_2t_4$ gives the n^{th} array of the language.

It should be noted that the sizes of the arrowheads B_1, B_2, B_3 are not fixed, but vary depending on the sizes of the hexagons in the input place of the transition, so that the condition for catenation is satisfied.

4 Comparison Results

In this section we recall the definition of hexagonal array languages [8], and hexagonal kolam array languages [7] and compare the generative power of HATPNS with $(X : Y)HAL$, $(R : Y)HKAL$ where $X, Y \in \{R, CF, CS\}$.

Definition 6. A Hexagonal Array Grammar (HAG) is $G = (N, I, T, P, S, \mathcal{L})$ where N, I, T are finite sets of non terminals, intermediates and terminals respectively; $P = P_1 \cup P_2$ is a finite set of productions and $S \in N$ is the start symbol. For each A in I , L_A is an intermediate language which is regular, CF or CS string language written in the appropriate arrowhead form. An arrowhead is written in the form $\{\dots \langle v \rangle \dots\}$ where $\langle v \rangle$ denotes the vertex and the arrowhead is written in the clockwise direction $\mathcal{L} = \{L_A/A \in I\}$. G is called $(X : R)HAG$, $(X : CG)HAG$ or $(X : CS)HAG$ depending on the rules of P_2 . X is R, CF or CS according as all intermediate languages are regular, atleast one of them is CF or atleast one of them is CS. The language generated by G is $(X : Y)HAL$.

Definition 7. A Hexagonal Kolam Array Grammar (HKAG) is G is a 5-tuple $(V, I, P, S, \mathcal{L})$ where $V = V_1 \cup V_2$, V_1 is a finite sets of non terminals and V_2 is a finite set of intermediates; I is a finite set of terminals; $P = P_1 \cup P_2$, P_1 is a finite set of non terminal rules of the form $S \rightarrow S_1 \begin{matrix} \nearrow \\ \circlearrowright \end{matrix} a$, $S \rightarrow S_1 \begin{matrix} \nwarrow \\ \circlearrowleft \end{matrix} b$, $S \rightarrow S_1 \begin{matrix} \downarrow \\ \circlearrowdown \end{matrix} c$ where $S, S_1 \in V_1$, $a, b, c \in V_2$ and P_2 is a terminal rule of the form $S \rightarrow H$ where $S \in V_1$ and H is a hexagonal array over I ; S is the start symbol; \mathcal{L} is a set of intermediate languages corresponding to each one of the intermediate in V_2 . These intermediate languages are regular, CF or CS string languages written in the appropriate arrowhead from. An arrowhead is written in the form $\{\dots \langle v \rangle \dots\}$ where $\langle v \rangle$ denotes the vertex and the arrowhead is written in the clockwise direction. A HKAG is called $(R : R)HKAG$, $(R : CF)HKAG$, $(R : CS)HKAG$ according as all the members of \mathcal{L} is regular, atleast one of \mathcal{L} is CF or atleast one of \mathcal{L} is CS.

Theorem 1. Every $(R : X)HAL$, for $X \in \{R, CF, CS\}$ can be generated by HATPNS.

Proof. Let G be the corresponding $(R : X)HAG$. Let $S \rightarrow H \circledast S'$ be the initial rule in P_1 and $\mathcal{L} = \{L_A/A \in I\}$ be the set of intermediate languages. Define for every L_A an arrowhead of similar type from A_1 to A_6 . The parameters of the arrowhead will depend on the parameters of the hexagon in the input place. So firing the transition with catenation rules as labels will catenate the arrowhead to the hexagon. Let $H \circledast A_1 \circledast \dots \circledast A_k \circledast \dots \circledast A_l \circledast \dots \circledast A_n$ be the string of arrowheads A_i and H , which derives the first array of the language. Let $A_k \circledast \dots \circledast A_l$ be the substring which applied m times gives the m^{th} array of the language. The steps for constructing the HATPNS to generate the $(R : X)HAL$ would depend on the values of k, l, n . For $k = 1$ we have (i) $k = l = n$, (ii) $k = l < n$, (iii) $k < l = n$ and (iv) $k < l < n$. Similarly for $k > 1$, we have (v) $k = l = n$, (vi) $k = l < n$, (vii) $k < l = n$ and (viii) $k < l < n$. We give the steps for construction of HATPNS when $k < l < n$, ($k = 1$). The other cases can be dealt similarly.

- Step 1. Let p, p' be two places with H as a token, T is a transition with input places p, p' and output places p, p_1 . The label of T is p . Let $i = 1$.
- Step 2. Let t_i be a transition with input place p_i and label $H \circledast A_i$. If $i < l$, then the output place is p_{i+1} and goto next step. If $i = l$, then the output place is p_k and goto step 4.
- Step 3. Let $i = i + 1$ and repeat step 2.
- Step 4. Let t_{l+1} be a transition with input place p_k and label $H \circledast A_{l+1}$. If $l + 1 = n$, then the output places are p', P and goto step 7. If $l + 1 < n$, then the output place is p_{l+1} and put $i = l + 2$.
- Step 5. Let t_i be a transition with input place p_{i-1} and label $H \circledast A_i$. If $i < n$, then the output place is p_i and goto next step. If $i = n$, then the output places are p', P and goto step 7.

- Step 6. Let $i = i + 1$ and repeat step 5.
- Step 7. $F = \{P\}$.
- Step 8. END. □

Corollary 1. *Every language in $(R, Y)HKAL$ with $Y \in \{R, CF, CS\}$ can be generated by a HATPNS.*

Proof. It is known that $(R, Y)HKAL \subsetneq (R, Y)HAL$ [8]. By Theorem 1, any $(R, Y)HAL$ can be generated by a HATPNS. Thus every language in $(R, Y)HKAL$ can be generated by a HATPNS. □

Theorem 2. *For $X \in \{CF, CS\}$, $Y \in \{R, CF, CS\}$ the family $(X : Y)HAL$ cannot be generated by HATPNS.*

Proof. In $(CF : Y)HAG$ the rules in P_2 would have a sequence of catenation on H a certain number of times and follow it by another sequence of catenations the same number of times. If the Petri net structure has a subnet C_1 for the first sequence of catenations and another subnet C_2 for the second sequence of catenations then there would be no control on the number of times C_1 and C_2 get executed. Hence HATPNS cannot generate a $(CF : Y)HAL$. Since $(CF : Y)HAL \subsetneq (CS : Y)HAL$, HATPNS cannot generate a $(CS : Y)HAL$. □

5 Adjunct Hexagonal Array Token Petri Net Structure

In this section we introduce a variation of the previous model to generate hexagonal pictures known as Adjunct Hexagonal Array Token Petri Net Structure (AHATPNS) and compare its generative power with HATPNS, controlled table 0L/1L hexagonal array grammars [7] and E2DHCFG [9].

For any hexagon H , $|H|_{LU}$ arrowheads of thickness one with same parameters can be found in the direction \nearrow (or its dual \swarrow). They are denoted by $lu_1, lu_2, \dots, lu_{|H|_{LU}}$. An A_3 type arrowhead (A_4 type) can be joined at any of the $|H|_{LU} + 1$ positions subject to the conditions of an arrowhead catenation.

The adjunction rule is a tuple $(H \begin{smallmatrix} \circlearrowright \\ \circlearrowleft \end{smallmatrix} A_3 / H \begin{smallmatrix} \circlearrowleft \\ \circlearrowright \end{smallmatrix} A_4, blu_i / alu_j), 1 \leq i, j \leq |H|_{LU}$, joining $A_3(A_4)$ into H either before lu_i or after lu_j .

$|H|_L$ arrowheads of thickness one with same parameters can be found in the direction \downarrow (or its dual \uparrow). They are denoted by $l_1, l_2, \dots, l_{|H|_L}$. An A_1 type arrowhead (A_2 type) can be joined at any of the $|H|_L + 1$ positions subject to the conditions of an arrowhead catenation. The adjunction rule is a tuple $(H \begin{smallmatrix} \circlearrowdown \\ \circlearrowup \end{smallmatrix} A_1 / H \begin{smallmatrix} \circlearrowup \\ \circlearrowdown \end{smallmatrix} A_2, bli_i / al_j), 1 \leq i, j \leq |H|_L$, joining $A_1(A_2)$ into H either before l_i or after l_j .

$|H|_{LL}$ arrowheads of thickness one with same parameters can be found in the direction \nwarrow (or its dual \searrow). They are denoted by $ll_1, ll_2, \dots, ll_{|H|_{LL}}$. An $A_5(A_6)$ type arrowhead can be joined at any one of $|H|_{LL} + 1$ positions subject to the conditions of arrowhead catenation. The adjunction rule is a tuple

$(H \begin{smallmatrix} \circlearrowleft \\ \circlearrowright \end{smallmatrix} A_5 / H \begin{smallmatrix} \circlearrowright \\ \circlearrowleft \end{smallmatrix} A_6, bli_i / all_j), 1 \leq i, j \leq |H|_{LL}$, joining $A_5(A_6)$ into H either before ll_i or after ll_j . Refer to Figs. 6 and 7 for positions in a hexagon.

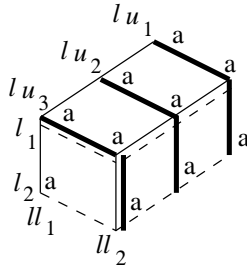


Fig. 6. Positions of adjunction of a hexagon in the directions ↗, ↖ and ↓

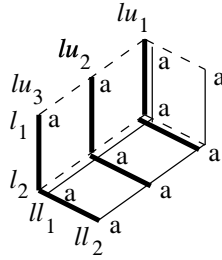


Fig. 7. The corresponding dual positions ↘, ↙, ↑

Note. In Figs. 6 and 7, l_i is represented by dotted lines, ll_i is represented by normal lines, lu_i is represented by thick lines.

5.1 Adjunction Rules as Labels

Let a transition t have $(H \circledast A, \text{position})$ a tuple as a label, where H is the hexagonal array in all its input places, A a predefined arrowhead, \circledast is one of the six directions and position being any one of the positions shown in Figs. 6 and 7. Then firing the transition will join A into H in the position given as the second argument of the tuple and the resulting hexagon is put in the output places. The conditions required for catenation should be satisfied, otherwise the transition cannot fire. An example explaining the adjunction rule is given below.

Example 3.

$$\text{Let } H = \begin{matrix} & & & & a \\ & & & & a & a \\ a & a & & & & \\ a & a & & & & \\ & & & & & a \end{matrix}, \quad A_1 = \begin{matrix} & & & & x & x \\ & & & & x & \\ x & x & & & & \\ & & & & x & \end{matrix}, \quad H_1 = \begin{matrix} & & & & a \\ & & & & a & a \\ & & & & a \\ x & x & & & & \\ x & x & & & & \\ & & & & & x \\ & & & & & a & a \\ & & & & & a \end{matrix}$$

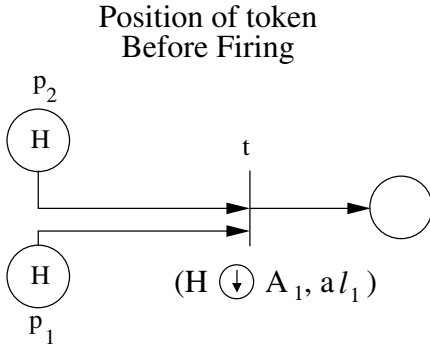


Fig. 8.

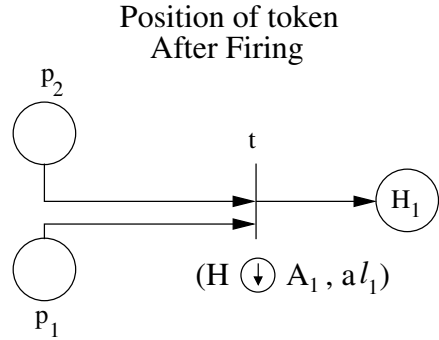


Fig. 9.

Definition 8. If $C = (P, T, I, O)$ is a Petri net structure with hexagonal arrays of Σ^{**H} as initial markings $M_0 : P \rightarrow \Sigma^{**H}$, labels of transitions being adjunct rules and a finite set of final places $F \subseteq P$, then the Petri net structure is defined as Adjunct Hexagonal Array Token Petri Net Structure (AHATPNS).

Definition 9. If C is a AHATPNS then the language generated by the Petri net C is defined as

$$L(C) = \{H \in \Sigma^{**H} / H \text{ is in } p \text{ for some } p \in F\}$$

With hexagonal arrays of Σ in some places as initial marking all possible sequences of transitions are fired. The set of all arrays collected in the final places F is called the language generated by C .

Example 4. Let $C_1 = \begin{pmatrix} y \\ x \end{pmatrix}^3 \langle \begin{matrix} y \\ x \end{matrix} \rangle \begin{pmatrix} y \\ x \end{pmatrix}^3$ and $F = \{p_2\}$.

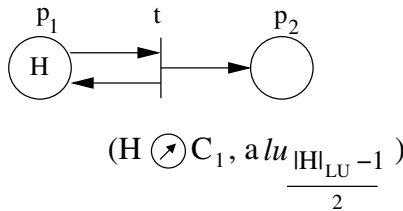


Fig. 10. AHATPNS

Firing t once yields

$$H = \begin{array}{ccc} & & b \\ & & y \ b \\ & b & x \ y \ b \\ a \ b & & a \ x \ y \ b \\ a \ a \ b & & a \ a \ x \ y \\ a \ a \ b & & a \ a \ x \ y \\ a \ a \ a & & a \ a \ x \ b \\ a \ a \ b \xrightarrow{t} & a \ a \ a \ y \\ a \ a \ a & & a \ a \ x \ b \\ a \ a \ b & & a \ a \ a \ y \\ a \ a \ a & & a \ a \ x \ b \\ a \ a \ a & & a \ a \ x \ y \\ a \ a \ b & & a \ a \ a \ y \\ a \ a & & a \ a \ x \\ a & & a \ a \\ & & a \end{array}$$

The firing sequence t^2 joins the arrowhead $\begin{pmatrix} y \\ y \\ x \\ x \end{pmatrix}^3 \langle \begin{matrix} y \\ x \\ x \end{matrix} \rangle \begin{pmatrix} y \\ y \\ x \\ x \end{pmatrix}^3$ into H .

Theorem 3. $HATPNS \subsetneq AHATPNS$.

Proof. Every arrowhead catenation rule is a special case of adjunction rule. The catenation rule $H \begin{matrix} \curvearrowright \\ \curvearrowright \end{matrix} A_3$ is equivalent to $(H \begin{matrix} \curvearrowright \\ \curvearrowright \end{matrix} A_3, blu_1)$. Similarly all catenation rules have an equivalent adjunction rule. Thus every HATPNL can be generated by some AHATPNS. The picture language generated by Example 4 cannot be generated by any HATPNS. Thus we have proper inclusion. \square

We are now ready to compare AHATPNS with other models. For this, we recall the following definitions.

Definition 10. A controlled table 0L hexagonal array model is a 4-tuple (V, \mathcal{P}, H_0, C) where V is a finite alphabet; \mathcal{P} is a finite set of tables $\{P_1, P_2, \dots, P_k\}$ each table consisting of right up, left up, down (or the duals) rules of the form $a \rightarrow bc, a, b, c \in V$; H_0 is the axiom; C is a control language which may be regular, context-free or context-sensitive.

By changing the rules in the tables to be context dependent we get hexagonal 1L array models with regular, CF or CS control.

Definition 11. An extended 2D hexagonal context-free grammar (E2DHCFG) is $G = (V, \Sigma, P_{ur}, P_{ul}, P_d, P_{ll}, P_{lr}, P_u, H_0)$ where V is a finite set of symbols; $\Sigma \subseteq V$ is the set of terminal symbols; $P_{ur} = \{t_{ur}(i) / 1 \leq i \leq m\}$; Each $t_{ur}(i)$ ($1 \leq i \leq m$) called a UR table, is a set of CF rules of the form $A \rightarrow \alpha, A \in V \setminus \Sigma, \alpha \in V^*$ such that for any two rules $A \rightarrow \alpha, B \rightarrow \beta$ we have $|\alpha| = |\beta|$ where $|\alpha|$

denotes the length of α ; Each of the other five components $P_{ul}, P_d, P_{ll}, P_{lr}, P_u$ is similarly defined; $H_0 \subseteq \Sigma^{h^{**}} \setminus \{\lambda\}$ is a finite set of axiom arrays that are hexagonal arrays.

Proposition 1. *The family AHATPNL is incomparable with the controlled table 0L/1L hexagonal array languages but not disjoint.*

Proof. The set of all p hexagons with alternate sides equal and of order 1 belong to T1LHAL with regular control [7]. This family can be generated by a AHATPNS. The set of all regular hexagons over a single letter with side 2^n can be generated by T0LHAG with CS control. This family cannot be generated by AHATPNS because a control on a firing sequence cannot be imposed in this model. Example [4] does not belong to T0LHAL since the development of the array is not along the edges. Thus the two families are incomparable but not disjoint. \square

Proposition 2. *E2DHCFL and AHATPNL are not mutually disjoint.*

Proof. Example [4] gives a picture language which belongs to both E2DHCFL [9] and AHATPNL. \square

6 Conclusion

Two models for generating hexagonal arrays have been defined and compared with some of the already existing models. These models are able to generate certain families of HAL. If some sort of control is defined on the sequence of firing, the other families of HAL can also be generated. It is worth examining the possibilities of defining a control over the firing sequences and obtain further results. It would also be interesting to define a general framework with arbitrary angles dividing equally a circle. Comparisons of controlled table 0L/1L hexagonal array models and E2DHCFL with HATPNS are kept for our future work.

References

1. Dersanambika, K.S., Krithivasan, K., Martin-Vide, C., Subramanian, K.G.: Local and recognizable hexagonal picture languages. *International Journal of Pattern Recognition and Artificial Intelligence* 19(7), 553–571 (2005)
2. Dersanambika, K.S., Krithivasan, K., Agarwal, H.K., Gupta, J.: Hexagonal contextual array P systems. *Formal Models, Languages and Application. Series in Machine Perception Artificial Intelligence* 66, 79–96 (2006)
3. Peterson, J.L.: *Petri net theory and modeling of systems*. Prentice-Hall, Englewood Cliffs (1981)
4. Lalitha, D., Rangarajan, K.: Adjunct array token Petri nets. In: *Proceedings of International Conference on Operations Research Applications in Engineering and Management (ICOREM)*, pp. 431–445. Anna University, Tiruchirapalli (2009)
5. Lalitha, D., Rangarajan, K.: Column and row catenation Petri net systems. In: *Proceeding of Fifth IEEE International Conference on Bio-Inspired Computing: Theories and Applications*, pp. 1382–1387 (2010)

6. Middleton, L., Sivaswamy, J.: Hexagonal Image Processing: A Practical Approach. Springer, Heidelberg (2005)
7. Siromoney, G., Siromoney, R.: Hexagonal arrays and rectangular blocks. *Computer Graphics and Image Processing* 5, 353–381 (1976)
8. Subramanian, K.G.: Hexagonal array grammar. *Computer Graphics and Image Processing* 10, 388–394 (1979)
9. Subramanian, K.G., Geethalakshmi, M., Atulya, K., Nagar, L.S.K.: Hexagonal picture languages. In: Proceedings of the 5th Asian Mathematical Conference, Malaysia, pp. 510–514 (2009)

Binary Images, M -Vectors, and Ambiguity

K.G. Subramanian¹, Kalpana Mahalingam²,
Rosni Abdullah¹, and Atulya K. Nagar³

¹ School of Computer Sciences, Universiti Sains Malaysia,
11800 Penang, Malaysia

² Department of Mathematics, Indian Institute of Technology Madras,
Chennai 600028 India

³ Department of Computer Science,
Liverpool Hope University, Liverpool L16 9JD UK

Abstract. Mateescu et al (2001) introduced the notion of Parikh matrix of a word as an extension of the well-known concept of Parikh vector of a word. The Parikh matrix provides more numerical information about a word than given by the Parikh vector. Here we introduce the notion of M -vector of a binary word which allows us to have a linear notation in the form of a unique vector representation of the Parikh matrix of the binary word. We then extend this notion of M -vector to a binary image treating it as a binary array over a two-symbol alphabet. This is done by considering the M -vectors of the words in the rows and columns of the array. Among the properties associated with a Parikh matrix, M -ambiguity or simply ambiguity of a word is one which has been investigated extensively in the literature. Here M -ambiguity of a binary array is defined in terms of its M -vector and we obtain conditions for M -ambiguity of a binary array.

1 Introduction

The Parikh mapping or the Parikh vector [14] of a word w over an alphabet Σ which enumerates the number of occurrences of the symbols of Σ in the word w , is a well-known important notion in formal language theory [15]. But many words can have the same Parikh vector and hence the Parikh mapping is not injective and so much information about a word is lost in the transition from words to vectors. Mateescu et al [13] introduced an extension of the Parikh vector, known as Parikh matrix mapping or simply referred to as Parikh matrix, based on a certain type of matrix, which gives more numerical information about a word than a Parikh vector does. For instance, in the case of a word over a binary alphabet $\Sigma = \{a, b\}$ with an ordering $a < b$, the Parikh matrix gives information on the number of a 's, the number of b 's (as in the Parikh vector), but also gives information on the number of subwords (also called scattered-subwords [7]) ab , which are subsequences a, b of length two. As an example, in the word $aabab$, there are three a 's, two b 's and five subwords ab . Although the Parikh matrix mapping is also not injective in general, two words with the same Parikh vector have in many cases different Parikh matrices. After the introduction of

this interesting notion of a Parikh matrix, there has been a series of studies [1,2,3,8,11,12,16,17,18,19,20,21,22] on this notion, especially on the injectivity problem or the complement problem, known as M -ambiguity of words.

On the other hand binary images are binary arrays over a two-letter alphabet, say, $\Sigma = \{a, b\}$. For example, interpreting a as a ‘white square’ and b as a ‘black square’, the binary images of the chess-board patterns in Figure 1, are binary arrays as shown in Figure 2.

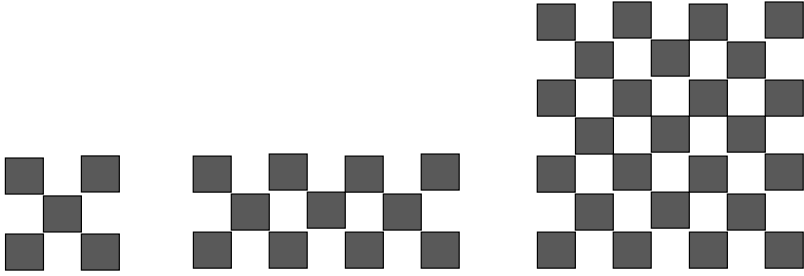


Fig. 1. Chessboard patterns

```

                b a b a b a b
                a b a b a b a
    b a b   b a b a b a b   b a b a b a b
    a b a   a b a b a b a   a b a b a b a
    b a b   b a b a b a b   b a b a b a b
                a b a b a b a
                b a b a b a b
    
```

Fig. 2. Binary arrays of binary images of Fig.1

There are many studies on various problems of interest related to binary images such as combinatorial properties [5,6], connectedness results [9], thinning methods [24], reconstruction problems [4,10] and many others.

Here we make a theoretical study by examining the notion of Parikh matrix in the context of a binary image considered as a binary array A . For doing this, we first introduce the notion of M -vector of a binary word. This enables us to have a linear notation in the form of a unique vector representation of the Parikh matrix of a binary word. The components of the M -vector of a binary word w give the number of a 's, the number of b 's and the number of subwords ab in w . We then extend this notion of M -vector to a binary image treating it as a binary array over a two-symbol alphabet. The M -vector of a binary array A gives the number of a 's and the number of b 's in A and the number of ‘horizontal’ subwords ab in the rows of A and the number of ‘vertical’ subwords $\begin{smallmatrix} a \\ b \end{smallmatrix}$ in the columns of A . The notion of M -ambiguity or simply ambiguity of a word and in particular, of

a binary word has been extensively investigated [1,2,3,8,11,13,17,21,22]. Here we introduce the notion of M -ambiguity of a binary array and obtain conditions for M -ambiguity of a binary array. Section 2 recalls the basic notions and results needed in the subsequent sections. Section 3 introduces the notion of M -vector of a binary image considered as a binary array. Section 4 introduces a notion of ambiguity of a binary array and obtains conditions for M -ambiguity of a binary array. The paper ends with a concluding section 5.

2 Preliminaries

Let Σ , called an alphabet, be a finite set of symbols. A word over Σ is a finite sequence of symbols from Σ . We denote by Σ^* the set of all words over Σ . The empty word is denoted by λ . For a word $w \in \Sigma^*$, $|w|$ denotes the length of w .

A word u is called a *subword* (also called scattered subword) of a word w , if there exist words x_1, \dots, x_n and y_0, \dots, y_n , (some of them possibly empty), such that $u = x_1 \cdots x_n$ and $w = y_0 x_1 y_1 \cdots x_n y_n$. For example if $w = aabbaabab$ is a word over the alphabet $\{a, b\}$, then $ababa$ is a subword of w . The number of occurrences of the word u as a subword of the word w is denoted by $|w|_u$. In particular, if u is a symbol in the alphabet, then $|w|_u$ equals the number of occurrences of the symbol u in w . Two occurrences of a subword are considered different if they differ by at least one position of some letter.

An ordered alphabet $\Sigma = \{a_1 < a_2 < \dots < a_k\}$ is an alphabet $\Sigma = \{a_1, \dots, a_k\}$ with the ordering $a_1 < a_2 < \dots < a_k$. The Parikh vector of a word w counts the number of occurrences of the symbols of the alphabet in the word w . In fact if $\Sigma = \{a_1, \dots, a_k\}$, then the Parikh vector of w is the vector $(|w|_{a_1}, \dots, |w|_{a_k})$.

We now recall the definition of a Parikh matrix mapping [13], which is a generalization of the Parikh mapping or Parikh vector [14]. A triangle matrix is a square matrix $M = (m_{i,j})_{1 \leq i,j \leq k}$, such that $m_{i,j}$ are non-negative integers for all $1 \leq i, j \leq k$, $m_{i,j} = 0$, for all $1 \leq j < i \leq k$, and, moreover, $m_{i,i} = 1$, for all $1 \leq i \leq k$. The set M_k of all triangle matrices of dimension $k \geq 1$ is a monoid with respect to multiplication of matrices.

Definition 1. [13] Let $\Sigma = \{a_1 < a_2 < \dots < a_k\}$ be an ordered alphabet. The Parikh matrix mapping is the monoid morphism $\Psi_k : \Sigma^* \rightarrow M_{k+1}$, defined by the condition: $\Psi_k(\lambda) = I_{k+1}$, the $(k + 1) \times (k + 1)$ unit matrix, and if $\Psi_k(a_q) = (m_{i,j})_{1 \leq i,j \leq (k+1)}$, then for each $1 \leq i \leq (k + 1)$, $m_{i,i} = 1, m_{q,q+1} = 1$, all other elements of the matrix $\Psi_k(a_q)$ are 0.

For a word $w = a_{i_1} a_{i_2} \cdots a_{i_m}, a_{i_j} \in \Sigma$ for $1 \leq j \leq m$, we have

$$\Psi_k(w) = \Psi_k(a_{i_1})\Psi_k(a_{i_2}) \cdots \Psi_k(a_{i_m}).$$

In other words $\Psi_k(w)$ is computed by multiplication of matrices and the triangle matrix $\Psi_k(w)$ is called the Parikh matrix of w .

For example, if $\Sigma = \{a < b\}$ and $w = aabab$, then

$$\begin{aligned} \Psi_2(a) &= \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \Psi_2(b) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}, \\ \Psi_2(aabab) &= \Psi_2(a)\Psi_2(a)\Psi_2(b)\Psi_2(a)\Psi_2(b) \\ &= \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 3 & 5 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{pmatrix} \end{aligned}$$

Note that $\Psi_2(w)$ is a 3×3 triangle matrix.

We now recall some important facts about Parikh matrices.

Lemma 1. [13] Let $\Sigma = \{a_1 < a_2 < \dots < a_k\}$ and $w \in \Sigma^*$. The Parikh matrix $\Psi_k(w) = (m_{i,j})_{1 \leq i,j \leq (k+1)}$, has the following properties : i) $m_{i,j} = 0$, for all $1 \leq j < i \leq (k+1)$, ii) $m_{i,i} = 1$, for all $1 \leq i \leq (k+1)$, iii) $m_{i,j+1} = |w|_{a_i a_{i+1} \dots a_{j-1} a_j}$, for all $1 \leq i \leq j \leq k$.

The Parikh matrix mapping is not injective. For example, if the alphabet is $\Sigma = \{a < b\}$, the words $w_1 = abbabbbb$ and $w_2 = ababbbabb$ have the same Parikh matrix, namely,

$$\begin{pmatrix} 1 & 3 & 13 \\ 0 & 1 & 6 \\ 0 & 0 & 1 \end{pmatrix}$$

The word w_1 (or the word w_2) is then called M -ambiguous or simply ambiguous. Many of the studies (see for example, [1,2,3,20,21,8,11,17]) in this area deal with this problem of M -ambiguity.

In the case of a binary alphabet $\Sigma = \{a < b\}$, characterizations of equality of Parikh matrices are known [8]. We recall one such characterization here.

For words $u, v \in \Sigma^*$, with $\Sigma = \{a < b\}$, define $u \equiv v$ if there exist words x, y, z such that $u = xabybaz, v = xbayabz, x, y, z \in \Sigma^*$. The relation \equiv is an equivalence relation [8].

Lemma 2. [1,8,11] For words u, v over Σ , the Parikh matrices $\Psi_2(u)$ and $\Psi_2(v)$ are equal if and only if $u \equiv v$.

A word over $\Sigma = \{a < b\}$ is said to be unambiguous if it is not ambiguous. A characterization [11] of unambiguous words over $\Sigma = \{a < b\}$ is known in terms of regular expressions [15]. We recall this here.

Lemma 3. [11] A word over $\{a, b\}, a < b$ is unambiguous if and only if it belongs to the language denoted by the regular expression

$$a^*b^* + b^*a^* + a^*ba^* + b^*ab^* + a^*bab^* + b^*aba^*$$

3 M–Vector of a Binary Array

We restrict our attention to an ordered binary alphabet $\Sigma = \{a < b\}$. A binary image or a binary $m \times n$ array A over the alphabet Σ is a rectangular arrangement of symbols from Σ in m rows and n columns. We will write such an array as

$$A = \begin{matrix} a_{11} & \cdots & a_{1n} \\ \cdots & & \cdots \\ a_{m1} & \cdots & a_{mn} \end{matrix}, \quad \text{and the transpose of } A, A^t = \begin{matrix} a_{11} & \cdots & a_{m1} \\ \cdots & & \cdots \\ a_{1n} & \cdots & a_{mn} \end{matrix},$$

$a_{ij} \in \Sigma$. We always consider $a_{11} \cdots a_{1n}$ as the first row of A .

$a \ b \ a \ b \ b$	$a \ a \ b \ b \ a$ is 3×5 binary array and its transpose is	$a \ a \ b$
$a \ a \ b \ b \ a$		$b \ a \ a$
$b \ a \ a \ b \ a$		$b \ b \ b$
		$b \ a \ a$

will call the words in the rows of a binary array as horizontal words or simply words and the words in the columns of a binary array as vertical words over Σ .

Thus a vertical word w over $\Sigma = \{a < b\}$ is of the form $w = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_m \end{pmatrix}$, $a_i \in \Sigma$ for

$i = 1, \dots, m$. For a word $x = b_1 b_2 \cdots b_n$, $b_i \in \Sigma$ for $i = 1, \dots, n$, we denote by x^t the vertical word $\begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$. Also we define $(x^t)^t = x$. The notion of a subword

in a vertical word is analogous to the notion of a subword in a word except that the subword itself is a vertical word.

The set of all binary arrays over Σ is denoted by Σ^{**} . We denote respectively by \circ and \diamond the *column concatenation* and *row concatenation* of arrays in Σ^{**} . In contrast to the case of strings, these operations are partially defined, namely, for any $A, B \in \Sigma^{**}$, $A \circ B$ is defined if and only if A and B have the same number of rows. Similarly $A \diamond B$ is defined if and only if A and B have the

$a \ b \ b \ a \ a$	$a \ a \ b$
$b \ a \ a \ b \ b$ and $B = a \ b \ b$, then	
$a \ b \ a \ b \ b$	$b \ b \ a$

$A \circ B = a \ b \ b \ a \ a \ a \ b$, where as the row catenation $A \diamond B$ is not defined, since $a \ b \ a \ b \ b \ b \ b \ a$

the number of columns in A and B are not equal.

We now introduce the notion of a M –vector of a binary word (a horizontal or a vertical word) that allows us to provide a linear notation in the form of a unique vector representation of the Parikh matrix of a binary word. The linear notation is an equivalent alternative form of the Parikh matrix of a binary word

and is a concise notation as well. A result that involves the Parikh matrix of a binary word can therefore be expressed in this notation.

Definition 2. Let $\Sigma = \{a < b\}$. A mapping M , called M -mapping, from Σ^* to N^3 , where $N = \{0, 1, 2, \dots\}$, is defined recursively as follows:

$$M(\lambda) = (0, 0, 0), M(a) = (1, 0, 0), M(b) = (0, 1, 0)$$

For $w_1, w_2 \in \Sigma^*$,

$$M(w_1 w_2) = M(w_1) \oplus M(w_2)$$

where

$$M(w_1) \oplus M(w_2) = (p_1 + p_2, q_1 + q_2, r_1 + r_2 + p_1 q_2),$$

if

$$M(w_1) = (p_1, q_1, r_1), M(w_2) = (p_2, q_2, r_2).$$

$M(w)$ is called the M -vector of the word $w \in \Sigma^*$.

The M -vector $M(w^t)$ of a vertical word w^t is defined as $M(w^t) = M(w)$.

For two words w_1, w_2 , we say that the M -vectors $M(w_1)$ and $M(w_2)$ commute with respect to \oplus , if $M(w_1 w_2) = M(w_2 w_1)$.

Remark

i) Note that the operator \oplus is associative.

ii) The M -vector $M(w) = (p, q, r), w \in \Sigma^*$, if and only if the Parikh matrix of w is

$$\Psi_2(w) = \begin{pmatrix} 1 & p & r \\ 0 & 1 & q \\ 0 & 0 & 1 \end{pmatrix}$$

iii) Also the Parikh matrix

$$\begin{aligned} \Psi_2(w_1 w_2) &= \Psi_2(w_1) \Psi_2(w_2) \\ &= \begin{pmatrix} 1 & p_1 + p_2 & r_1 + r_2 + p_1 q_2 \\ 0 & 1 & q_1 + q_2 \\ 0 & 0 & 1 \end{pmatrix} \end{aligned}$$

if and only if the M -vector

$$M(w_1 w_2) = M(w_1) \oplus M(w_2) = (p_1 + p_2, q_1 + q_2, r_1 + r_2 + p_1 q_2).$$

We now extend the notion of M -vector to a binary array.

Definition 3. Given a binary $m \times n$ array A over $\Sigma = \{a < b\}$, and denoting the words in the m rows of A by x_1, \dots, x_m and the vertical words in the n columns of A by y_1, \dots, y_n so that

$$A = x_1 \diamond x_2 \cdots \diamond x_m = y_1 \circ y_2 \cdots \circ y_n,$$

let the M -vectors of the words in the rows and the vertical words in the columns of A be respectively $M(x_i), 1 \leq i \leq m$ and $M(y_j^t), 1 \leq j \leq n$. If $\Sigma_{i=1}^m M(x_i) = (p, q, r)$ and $\Sigma_{j=1}^n M(y_j^t) = (p, q, s)$, (where the M -vectors are added componentwise), then the M -vector $M(A)$ of the binary array A is defined as

$$M(A) = (p, q, r, s).$$

Remark. Note that the entries p, q in the M -vector $M(A)$ of a binary array A are respectively, the number of a 's and the number of b 's in the binary array A whereas the entries r, s are respectively, the number of subwords ab in the rows and the number of subwords $\begin{smallmatrix} a \\ b \end{smallmatrix}$ in the columns of the binary array A .

Example 1. Consider the binary array

$$A_1 = \begin{matrix} a & b & a & a & b \\ a & a & b & b & a \\ b & a & b & a & b \\ b & b & a & b & a \\ a & a & b & a & b \end{matrix}$$

The M -vectors $M(x_i), 1 \leq i \leq 5$ of the words

$$x_1 = abaab, x_2 = aabba, x_3 = babab, x_4 = bbaba, x_5 = aabab$$

in the rows are respectively $(3, 2, 4), (3, 2, 4), (2, 3, 3), (2, 3, 1), (3, 2, 5)$.

The M -vectors $M(y_j^t), 1 \leq j \leq 5$ of the vertical words

$$y_1 = \begin{matrix} a \\ a \\ b \\ a \end{matrix}, y_2 = \begin{matrix} b \\ a \\ b \\ a \end{matrix}, y_3 = \begin{matrix} a \\ b \\ a \\ b \end{matrix}, y_4 = \begin{matrix} a \\ b \\ b \\ a \end{matrix}, y_5 = \begin{matrix} b \\ a \\ a \\ b \end{matrix}$$

in the columns are respectively $(3, 2, 4), (3, 2, 2), (2, 3, 4), (3, 2, 3), (2, 3, 3)$ so that

$$\Sigma_{i=1}^5 M(x_i) = (13, 12, 17), \Sigma_{j=1}^5 M(y_j) = (13, 12, 16).$$

Hence the M -vector of A_1 is $(13, 12, 17, 16)$.

4 Ambiguity of a Binary Array

Two $m \times n$ binary arrays A, B over $\Sigma = \{a < b\}$ are said to be M -equivalent if $M(A) = M(B)$ and in this case we say that A (as well as B) is called M -ambiguous or simply ambiguous. A binary array A is called unambiguous if it is not ambiguous.

Example 2. The binary array A_1 in example 1 is M -ambiguous since $M(A_1) = M(A_2) = (13, 12, 17, 16)$ for

$$\begin{array}{c}
 a a b b a \\
 a b a a b \\
 A_2 = b a b a b \\
 b b b a a \\
 a a a b b
 \end{array}$$

Lemma 4. A binary array A may be ambiguous even if the words in all the rows and the vertical words in all the columns of the binary array A are unambiguous.

Proof. Consider the binary array

$$\begin{array}{c}
 a a a b b \\
 a b a b b \\
 A_3 = a a a b b \\
 a a a a b \\
 b a b a a
 \end{array}$$

By Lemma 3, the words in the rows and the vertical words in the columns are all unambiguous. But the binary array A_3 is ambiguous since the M -vector of A_3 is $(15, 10, 22, 9)$ which is also the M -vector of the binary array A_4

$$\begin{array}{c}
 a a a b b \\
 a a b b b \\
 A_4 = a a a b b \\
 a a a a b \\
 b b a a a
 \end{array}$$

□

Given a binary array $A = \begin{matrix} a_{11} & \cdots & a_{1n} \\ \cdots & & \cdots \\ a_{m1} & \cdots & a_{mn} \end{matrix}$, over a binary alphabet Σ , a 2×2 sub-

array W of A is of the form $W = \begin{matrix} a_{ij} & a_{ik} \\ a_{lj} & a_{lk} \end{matrix}$, for some $i, j, k, l, 1 \leq i, l \leq m, 1 \leq k, j \leq n, i < l, j < k$. The subarray W is said to occur in A .

Lemma 5. In a binary array A , if the 2×2 subarray $W_1 = \begin{matrix} a & b \\ b & a \end{matrix}$ occurs and is replaced by the 2×2 subarray $W_2 = \begin{matrix} b & a \\ a & b \end{matrix}$ or vice versa, to obtain another binary array B , then the M -vectors of A and B are the same.

Proof. If the 2×2 subarray W_1 occurs in A , then there are two rows, say, rows i and l in A , with the row i containing the letter a in column j and the letter b in column k and with row l containing the letter b in column j and the letter a in column k . Replacing in A the subarray W_1 by W_2 , the number of

subword ab in the M -vector of the word in row i of A , will decrease by $k - j$ and in row l will increase by $k - j$. Hence the number of subword ab does not change in the sum of all the M -vectors of the words in all the rows of A . Also the number of a 's and the number of b 's in A do not change by this replacement. Similarly for the M -vectors of the vertical words in the columns of A . Hence the M -vector of B is the same as that of A . \square

We now obtain a sufficient condition for M -ambiguity of a binary array.

Theorem 1. A binary array A is M -ambiguous, if either the 2×2 subarray W_1 or the 2×2 subarray W_2 as in Lemma 5, occurs in A .

The theorem follows from Lemma 5. \square

Remark. Theorem 1 provides a sufficient condition for the M -ambiguity of a binary array. But this condition is not a necessary condition for M -ambiguity of a binary array. In fact the binary array

$$A = \begin{matrix} a & a & a \\ a & b & b \\ a & b & a \end{matrix}$$

is M -ambiguous since the binary array A and the binary array

$$B = \begin{matrix} a & b & a \\ a & a & b \\ b & a & a \end{matrix}$$

have the same M -vector but neither W_1 nor W_2 as in Theorem 1 occurs in A .

Let w be a given binary word. Then for the M -vector of the word w , $M(w) = (p, q, r)$, define $\rho(w) = (p, q)$.

Theorem 2. Let A_1 and A_2 be two $m \times n$ binary arrays and let x_{1i} 's be the words in the rows of A_1 and x_{2i} 's the words on the rows of A_2 and similarly let y_{1j} 's and y_{2j} 's be the vertical words in the columns of A_1 and A_2 respectively. Let A_1 and A_2 be such that $\rho(x_{1i}) = \rho(x_{2i})$ and $\rho(y_{1j}^t) = \rho(y_{2j}^t)$ for all $1 \leq i \leq m$ and $1 \leq j \leq n$. Then A_1 and A_2 are M -equivalent if and only if atleast one of the following holds: *i*) the subarray $W_1 = \begin{matrix} a & b \\ b & a \end{matrix}$ occurs in A_1 and A_2 is obtained

from A_1 by replacing W_1 by the subarray $W_2 = \begin{matrix} b & a \\ a & b \end{matrix}$ or *ii*) the subarray W_1 occurs in A_2 and A_1 is obtained from A_2 by replacing W_1 by the subarray W_2 .

The theorem follows from Lemma 5. \square

The notion of a ‘‘weak ratio-property’’ of words has been considered in [22]. We recall this notion for binary words and simply refer to it as ratio-property. Two binary words w_1, w_2 over $\Sigma = \{a < b\}$ are said to satisfy the ratio property, if the M -vectors $M(w_1) = (p_1, q_1, r_1), M(w_2) = (p_2, q_2, r_2), p_2 \neq 0$ and $q_2 \neq 0$ satisfy the following equality of the ratios:

$$\frac{p_1}{p_2} = \frac{q_1}{q_2}.$$

We then write $w_1 \sim_r w_2$. The relation \sim is indeed an equivalence relation. A result on Parikh matrices of words satisfying the ratio-property which has been shown in [22], is expressed here for binary words in terms of the M -vectors.

- Lemma 6.** For any two binary words w_1, w_2 over $\Sigma = \{a < b\}$ we have
- i)* $M(w_1w_2) = M(w_2w_1)$ if and only if $w_1 \sim_r w_2$. In other words w_1 and w_2 satisfy the ratio property if and only if the M -vectors $M(w_1)$ and $M(w_2)$ commute with respect to \oplus .
 - ii)* $M(w_1^n w_2^n) = M((w_1w_2)^n)$, for $n \geq 1$.

Proof. Let $M(w_1) = (p_1, q_1, r_1), M(w_2) = (p_2, q_2, r_2)$. Then

$$M(w_1w_2) = M(w_1) \oplus M(w_2) = (p_1 + p_2, q_1 + q_2, r_1 + r_2 + p_1q_2)$$

and

$$M(w_2w_1) = M(w_2) \oplus M(w_1) = (p_2 + p_1, q_2 + q_1, r_2 + r_1 + p_2q_1).$$

Hence $M(w_1w_2) = M(w_2w_1)$ if and only if the M -vectors satisfy $p_1q_2 = p_2q_1$. This proves *i)*. The statement *ii)* follows by repeatedly using *i)*. □

We now extend the notion of ratio-property to binary arrays.

Definition 5. Let A be a $m \times n$ binary array and B be another $m \times l$ binary array both over the alphabet $\Sigma = \{a < b\}$. Let the words in the m rows of A in order be $x_i, 1 \leq i \leq m$ and in the m rows of B in order be $y_i, 1 \leq i \leq m$. Let $M(x_i) = (p_{i1}, q_{i1}, r_{i1})$ and $M(y_i) = (p_{i2}, q_{i2}, r_{i2})$ for $1 \leq i \leq m$. Let $x_i \sim_r y_i$ for all $i, 1 \leq i \leq m$, so that

$$\frac{p_{i1}}{p_{i2}} = \frac{q_{i1}}{q_{i2}} = k_i, 1 \leq i \leq m,$$

where k_i is a constant. Then the binary arrays A and B are said to satisfy row ratio-property. We can analogously define a column ratio-property by considering the vertical words in the columns of A and B , but requiring A to be a $m \times n$ binary array and B be another $l \times n$ binary array.

Lemma 7 If $p_{i1} = k_i p_{i2}$ and $q_{i1} = k_i q_{i2}$ for $1 \leq i \leq m$, as in Definition 5, then $k_i = k_j$ for all $i, j, 1 \leq i, j \leq m$.

In view of lemma 7, we can take in Definition 5, $k_i = k$ for all $i, 1 \leq i \leq m$, where k is a constant.

Theorem 3. Let A be a $m \times n$ binary array and B be an $m \times l$ binary array both over the alphabet $\Sigma = \{a < b\}$. Then

- i)* $M(A \circ B) = M(B \circ A)$ if A and B satisfy row ratio-property.
- ii)* $M(A^n \circ B^n) = M((A \circ B)^n)$ where $A^n = A \circ A \circ \dots \circ A$.

Proof. The words in the rows of $A \circ B$ are $x_i y_i, 1 \leq i \leq m$ and in the rows of $B \circ A$ are $y_i x_i, 1 \leq i \leq m$. Suppose A and B satisfy row ratio-property. Then $x_i \sim_r y_i$, and so we have by Lemma 6, $M(x_i y_i) = M(y_i x_i)$ for each $i, 1 \leq i \leq m$ so that the sum of the M -vectors of the words in the rows of $A \circ B = \sum_{i=1}^m M(x_i y_i) = \sum_{i=1}^m M(y_i x_i) =$ the sum of the M -vectors of the words in the rows of $B \circ A$. Also the vertical words in the columns of $A \circ B$ are the same as the vertical words of the columns of $B \circ A$ although in a different order so that the sum of the M -vectors of the vertical words in both $A \circ B$ and $B \circ A$ is the same. Hence $M(A \circ B) = M(B \circ A)$. The second statement follows by repeated application of i . \square

Corollary. The binary array $A \circ B$ (as well as $B \circ A$) is M -ambiguous if A and B satisfy the row ratio-property.

We could consider in Theorem 3, the column-ratio property in the vertical words of the columns of A and B and formulate a corresponding result using the row catenation \diamond instead of the column catenation. We state this in the following Theorem.

Theorem 4. Let A be a $m \times n$ binary array and B be an $l \times n$ binary array both over the alphabet $\Sigma = \{a < b\}$. Then

- $i)$ $M(A \diamond B) = M(B \diamond A)$, if A and B satisfy column ratio-property.
- $ii)$ $M(A^n \diamond B^n) = M((A \diamond B)^n)$ where $A^n = A \diamond A \diamond \dots \diamond A$.

Corollary. The binary array $A \diamond B$ (as well as $B \diamond A$) is M -ambiguous if A and B satisfy the row-ratio property.

Remark. Unlike the case of strings, row (respy. column) ratio-property for two binary arrays with more than one row (respy. column), is only a sufficient condition for $A \circ B$ and $B \circ A$ (respy. $A \diamond B$ and $B \diamond A$) to be ambiguous. For example, if

$$\begin{array}{rcc}
 & a b a b & a a b a b a b a \\
 A = & a b a a & \text{and } B = a a a b a b a a \\
 & b a b a & a a b a b b b a \\
 & a a a b & a a a b a b b a
 \end{array}$$

then A and B do not satisfy row ratio-property but $A \circ B$ and $B \circ A$ have the same M -vector (30, 18, 75, 16) and so are ambiguous. Likewise A^t and B^t do not satisfy column ratio-property but $A \diamond B$ and $B \diamond A$ have the same M -vector (30, 18, 16, 75) and so are ambiguous.

5 Conclusion

We have introduced here the notion of M -vector of a binary word which is an equivalent linear representation of the Parikh matrix of the binary word. We then extended this notion of M -vector to a binary image considered as a

binary array. The notion of M -ambiguity of a binary array A is introduced here by requiring the existence of another binary array B of the same size as A and having the same M -vector as that of A . The notion of M -vector of a binary array has been used in the problem of reconstruction of binary images [23]. Possible extension of other properties of Parikh matrix of a word could be examined in the context of the M -vector of a binary array.

Acknowledgement. The authors are grateful to the referees for their useful comments.

References

1. Atanasiu, A.: Binary amiable words. Intern. J. Found. Comput. Sci. 18, 387–400 (2007)
2. Atanasiu, A., Martin-Vide, A.C., Mateescu, A.: On the injectivity of the Parikh matrix mapping. Fund. Inform. 46, 1–11 (2001)
3. Atanasiu, A., Atanasiu, R., Petre, I.: Parikh matrices and amiable words. Theor. Comput. Sci. 390, 102–109 (2008)
4. Balázs, P.: Discrete tomographic reconstruction of binary images with disjoint components using shape information. Int. J. Shape Modeling 14(2), 189–207 (2008)
5. Brimkov, V.E., Barneva, R.P.: Plane digitization and related combinatorial problems. Discrete Appl. Math. 147, 169–186 (2005)
6. Brimkov, V.E., Barneva, R.P.: Combinatorial approach to image analysis. Discrete Appl. Math. 157(16), 3359–3361 (2009)
7. Fazekas, S.Z., Nagy, B.: Scattered subword complexity of non-primitive words. J. Automata, Languages and Combinatorics 13, 233–247 (2008)
8. Fosse, S., Richomme, G.: Some characterizations of Parikh matrix equivalent binary words. Inf. Process. Letters 92, 77–82 (2004)
9. Gara, M., Tasi, T.S., Balázs, P.: Learning connectedness and convexity of binary images from their projections. Pure Math. Appl. 20, 27–48 (2009)
10. Masilamani, V., Krithivasan, K., Subramanian, K.G., Miin Huey, A.: Efficient Algorithms for Reconstruction of 2D-Arrays from Extended Parikh Images. In: Bebis, G., Boyle, R., Parvin, B., Koracin, D., Remagnino, P., Porikli, F., Peters, J., Klosowski, J., Arns, L., Chun, Y.K., Rhyne, T.-M., Monroe, L. (eds.) ISVC 2008, Part II. LNCS, vol. 5359, pp. 1137–1146. Springer, Heidelberg (2008)
11. Mateescu, A., Salomaa, A.: Matrix indicators for subword occurrences and ambiguity. Int. J. Found. Comput. Sci. 15, 277–292 (2004)
12. Mateescu, A., Salomaa, A., Yu, S.: Subword histories and Parikh matrices. J. Comp. System Sci. 68, 1–21 (2004)
13. Mateescu, A., Salomaa, A., Salomaa, K., Yu, S.: A Sharpening of the Parikh Mapping. Theoret. Informatics Appl. 35, 551–564 (2001)
14. Parikh, R.J.: On context-free languages. J. Assoc. Comput. Mach. 13, 570–581 (1966)
15. Salomaa, A.: Formal Languages. Academic Press, New York (1973)
16. Salomaa, A.: Subword histories and associated matrices. Theor. Comput. Sci. 407, 233–247 (2008)
17. Salomaa, A.: On the injectivity of Parikh matrix mappings. Fundam. Inf. 64, 391–404 (2005)

18. Salomaa, A., Yu, S.: Subword occurrences, Parikh matrices and Lyndon images. *Int. J. Found. Comput. sci.* 21, 91–111 (2010)
19. Serbanuta, T.F.: Extending Parikh matrices. *Theor. Comput. Sci.* 310, 233–246 (2004)
20. Serbanuta, V.N., Serbanuta, T.F.: Injectivity of the Parikh matrix mappings revisited. *Fundam. Inf.* 73, 265–283 (2006)
21. Serbanuta, V.N.: On Parikh Matrices, Ambiguity, and Prints, *Intern. J. Found. Comput. Sci.* 20, 151–165 (2009)
22. Subramanian, K.G., Miin Huey, A., Nagar, A.K.: On Parikh Matrices. *Int. J. Found. Comput. Sci.* 20(2), 211–219 (2009)
23. Subramanian, K.G., Isawasan, P., Abdullah, R.: Construction of a Binary Image with some prescribed Numerical Information (submitted for publication)
24. Wiederhold, P., Morales, S.: Thinning on cell complexes from polygonal tilings. *Discrete Appl. Math.* 157(16), 3424–3434 (2009)

Shuffle on Trajectories over Finite Array Languages

H. Geetha¹, D.G. Thomas², T. Kalyani¹, and A.S. Prasanna Venkatesan³

¹ Department of Mathematics, St. Joseph's College of Engineering
Chennai - 600119

² Department of Mathematics, Madras Christian College
Chennai - 600059

dgthomasmcc@yahoo.com

³ Department of Mathematics, B.S. Abdur Rahman University
Chennai - 600 048, India

Abstract. Parallel composition of words and languages of words appear as a fundamental operation in parallel computation and in the theory of concurrency. Shuffle on trajectories provides a method to handle the operation of parallel composition. In this paper we study the impact of shuffle on trajectories on various finite arrays. We obtain interesting results yielding new pictures and patterns. We compare this model with the other existing models.

Keywords: Trajectories, shuffle, array languages.

1 Introduction

Theoretical models for generating 2-dimensional arrays were proposed in [3]. These models describe a wide variety of interesting classes of pictures. Recently the interesting tool for the generation of languages of finite words is the shuffle on trajectories [8]. The operation of parallel composition leads to new shuffle like operations defined by syntactic constraints in the usual shuffle operation. The approach is applicable to concurrency, providing a method to define parallel composition. This operation is introduced using a uniform method based on the notion of trajectory. A trajectory is a segment of a line in plane, starting in the origin of axes and continuing parallel with the axis OX or OY. The line can change its direction only in points of non-negative integer coordinates. A trajectory defines how to skip from a word to another word during the shuffle operation. Shuffle on trajectories provides a method of great flexibility to handle the operation of parallel composition of processes from the catenation to the usual shuffle of processes. A fundamental study of the shuffle operation on finite words with trajectories has been made in [8]. The shuffle operation on finite arrays with trajectories has been introduced in [4] to develop the study on parallel contextual array grammars [5]. Based on the studies for generating various matrix array grammars [6] and on the notion of rewriting rules on various array

grammars [7], we use the shuffle on trajectories in this paper as a tool for generating various array languages. We present some interesting properties of shuffle on trajectories yielding nice patterns and pictures.

The paper is organized as follows: In section 2 we review necessary notions related to arrays, trajectories and shuffle on trajectories. In section 3, we introduce an array grammar with shuffle on trajectories and compare its generative power with that of other array grammars. We have made use of finite state matrix automata, pushdown matrix automata and online tessellation automata to obtain the language theoretic aspect of $L_1 \sqcup \sqcup_T L_2$ when L_1, L_2 are taken from different classes of array languages and T is either a regular language or a context-free language.

2 Preliminaries

In this section we recall some necessary notions and definitions for the study from [3,4,5,6,7,8].

Definition 1. Let Σ be a finite alphabet of symbols. A picture A over Σ is a rectangular $m \times n$ array of elements of the form

$$A = \begin{matrix} a_{m1} & \dots & a_{mn} \\ \vdots & \ddots & \vdots \\ a_{11} & \dots & a_{1n} \end{matrix} = [a_{ij}]_{m \times n}$$

The set of all pictures or arrays over Σ is denoted by Σ^{**} . A picture or an array language over Σ is a subset of Σ^{**} .

Definition 2. Let $A = [a_{ij}]_{m \times p}$, $B = [a_{ij}]_{n \times q}$. The column concatenation $A \oplus B$ of A and B is defined only when $m = n$ and is given by

$$A \oplus B = \begin{matrix} a_{m1} & \dots & a_{mp} & b_{n1} & \dots & b_{nq} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ a_{11} & \dots & a_{1p} & b_{11} & \dots & b_{1q} \end{matrix}$$

Similarly, the row concatenation $A \ominus B$ of A and B is defined only when $p = q$ and is given by

$$A \ominus B = \begin{matrix} & b_{n1} & \dots & b_{nq} \\ & \vdots & \ddots & \vdots \\ b_{11} & \dots & b_{1q} \\ a_{m1} & \dots & a_{mp} \\ & \vdots & \ddots & \vdots \\ a_{11} & \dots & a_{1p} \end{matrix}$$

The empty array is denoted by Λ , $\Lambda \oplus P = P \oplus \Lambda = P$ and $\Lambda \ominus P = P \ominus \Lambda = P$ for any $P \in \Sigma^{**}$.

Notations Σ^* denotes the set of all horizontal sequences of letters from Σ and $\Sigma^+ = \Sigma^* - \{\epsilon\}$, where ϵ is the identity element (of length zero). Σ_* denotes the set of all vertical sequences of letters over Σ and $\Sigma_+ = \Sigma_* - \{\epsilon\}$.

Definition 3. The column shuffle operation on two arrays P and Q denoted by $P \sqcup^c Q$ is defined recursively by

$$\begin{aligned} P \sqcup^c Q &= ((A \oplus X) \sqcup^c (B \oplus Y)) \\ &= A \oplus (X \sqcup^c (B \oplus Y)) \cup B \oplus ((A \oplus X) \sqcup^c Y) \end{aligned}$$

where $P = A \oplus X$ and $Q = B \oplus Y$, $P, Q \in \Sigma^{**}$, A is the first column of P and B is the first column of Q . The operation is defined only when the number of rows in P and the number of rows in Q are equal. If A is empty then $X = P$. Similarly if B is empty then $Y = Q$. Also $P \sqcup^c \Lambda = \Lambda \sqcup^c P = P$.

Definition 4. The row shuffle operation on two arrays P and Q denoted by $P \sqcup^r Q$ is defined recursively by

$$\begin{aligned} P \sqcup^r Q &= ((A \ominus X) \sqcup^r (B \ominus Y)) \\ &= A \ominus (X \sqcup^r (B \ominus Y)) \cup B \ominus ((A \ominus X) \sqcup^r Y) \end{aligned}$$

where $P = A \ominus X$ and $Q = B \ominus Y$, $P, Q \in \Sigma^{**}$, A is the first row of P and B is the first row of Q . The operation is defined only when the number of columns in P and the number of columns in Q are equal. Also $P \sqcup^r \Lambda = \Lambda \sqcup^r P = P$.

Definition 5. Let $V_1 = \{r, u\}$, $V_2 = \{\ell, d\}$ be the sets of versors in the plane where ℓ, r, u and d stands for the left, right, up and down directions respectively. A trajectory is an element $t \in V_1^* \cup V_2^*$.

Definition 6. The column shuffle of P with Q on the trajectory vt , $v \in \{r, u\}$, $t \in V_1^*$, is defined as

$$P \sqcup_{vt} Q = (A \oplus X) \sqcup_{vt} (B \oplus Y) = \begin{cases} A \oplus (X \sqcup_t (B \oplus Y)), & \text{if } v = r \\ B \oplus ((A \oplus X) \sqcup_t Y), & \text{if } v = u \end{cases}$$

$$\text{If } P = \Lambda, \Lambda \sqcup_{vt} (B \oplus Y) = \begin{cases} \phi & \text{if } v = r \\ B \oplus (\Lambda \sqcup_t Y) & \text{if } v = u \end{cases}$$

$$\text{If } Q = \Lambda, (A \oplus X) \sqcup_{vt} \Lambda = \begin{cases} A \oplus (X \sqcup_{vt} \Lambda) & \text{if } v = r \\ \phi & \text{if } v = u \end{cases}$$

$$\text{and } \Lambda \sqcup_{vt} \Lambda = \begin{cases} \Lambda & \text{if } t = \lambda \\ \phi & \text{otherwise.} \end{cases}$$

$$\text{If } T \subseteq V_1^*, \text{ then } P \sqcup_T Q = \bigcup_{t \in T} P \sqcup_t Q.$$

The row shuffle of P with Q on the trajectory vt , $v \in \{\ell, d\}$, $t \in V_2^*$ is defined in a similar way with r, u replaced by ℓ, d . Also if $|P|_c \neq |t|_r$ or $|Q|_c \neq |t|_u$ then $P \sqcup_t Q = \phi$. Similarly if $|P|_r \neq |t|_\ell$ or $|Q|_r \neq |t|_d$ then $P \sqcup_t Q = \phi$.

$$\text{If } T \subseteq V_2^*, \text{ then } P \sqcup_T Q = \bigcup_{t \in T} P \sqcup_t Q.$$

Example 1. Let P, Q and $R \in \Sigma^{**}$ be $P = \begin{matrix} a a a \\ a a a \end{matrix}$, $Q = \begin{matrix} b b b \\ b b b \end{matrix}$ and $R = \begin{matrix} c c c \\ c c c \end{matrix}$.

Then the shuffle on trajectory is given by

$$P \sqcup_t Q = \begin{matrix} a b b a b a \\ a b b a b a \\ a b b a b a \end{matrix}, \text{ where } t = ru^2rur. \text{ and}$$

$$P \sqcup_t R = \begin{matrix} a a a \\ c c c \\ a a a \\ c c c \\ c c c \\ a a a \end{matrix} \text{ where } t = ld^2ldl.$$

We now recall the notions of Siromoney matrix grammars [6] and Siromoney array grammars [7].

Definition 7. A Siromoney matrix grammar is a 2-tuple (G_1, G_2) , where $G_1 = (V_1, I_1, P_1, S)$ is a regular or context-free or context-sensitive or phrase-structure grammar where V_1 is a finite set of horizontal non-terminals. $I_1 = \{S_1, \dots, S_k\}$ a finite set of intermediates, $V_1 \cap I_1 = \phi$. P_1 is a finite set of production rules called horizontal production rules. $S \in V_1$ is the start symbol. $G_2 = (G_{21}, G_{22}, \dots, G_{2k})$ where $G_{2i} = (V_{2i}, T, P_{2i}, S_i)$, $1 \leq i \leq k$ are regular grammars, V_{2i} is a finite set of vertical non-terminals, $V_{2i} \cap V_{2j} = \phi$, $i \neq j$, T is a finite set of terminals, P_{2i} is a finite set of right linear production rules of the form $X \rightarrow aY$ or $X \rightarrow a$ where $X, Y \in V_{2i}$, $a \in T$, $S_i \in V_{2i}$, is the start symbol of G_{2i} .

The grammar G is called regular, context-free, context-sensitive and phrase-structure Siromoney matrix grammar if G_1 is regular, context-free, context-sensitive and phrase-structure respectively.

Derivations are defined as follows: First a string $S_{i1}S_{i2} \dots S_{in} \in I_1^*$ is generated horizontally using the horizontal production rules of P_1 in G_1 . i.e., $S \Rightarrow S_{i1}S_{i2} \dots S_{in} \in I_1^*$.

Vertical derivations proceed as follows: We write

$$\begin{matrix} A_{i1} \dots A_{in} \\ \Downarrow \\ B_{i1} \dots B_{in} \\ a_{i1} \dots a_{in} \end{matrix}$$

if $A_{ij} \rightarrow a_{ij}B_{ij}$ are rules in P_{2j} , $1 \leq j \leq n$. The derivation terminates if $A_{ij} \rightarrow a_{mj}$ are terminal rules in G_2 .

The set $L(G)$ of arrays generated by G consists of all $m \times n$ arrays $[a_{ij}]$ such that $1 \leq i \leq m$, $1 \leq j \leq n$ and $S_{\vec{G}_1}^* S_{i_1} S_{i_2} \dots S_{i_n} S_{\vec{G}_2}^* [a_{ij}]$.

$L(G)$ is called a phrase-structure matrix language (PSML), context-sensitive matrix language (CSML), context-free matrix language (CFML), regular matrix language (RML) if G is PSMG, CSMG CFMG, RMG respectively.

Definition 8. Let $G = (V, I, P, S)$ be an array (rewriting) grammar (AG), where $V = V_1 \cup V_2$, V_1 is a finite set of non-terminals, V_2 a finite set of intermediates, I a finite set of terminals, $P = P_1 \cup P_2 \cup P_3$, P_1 is the finite set of non-terminal rules, P_2 the finite set of intermediate rules and P_3 the finite set of terminal rules, $S \in V_1$ is the start symbol, P_1 is a finite set of ordered pairs (u, v) , $u, v \in (V_1 \cup V_2)^+$ or $u, v \in (V_1 \cup V_2)_+$.

An array grammar (AG) is called (CS : CS)AG if the non-terminal rules are CS and at least one intermediate language is CS. An array grammar is called (CS : CF)AG if the non-terminal rules are CS and none of the intermediate language is CS. A grammar is called (CS : R)AG if the non-terminal rules are CS and all the intermediate languages are regular. Similarly for all the other six, combinations namely, (CF : CS)AG, (CF : CF)AG, (CF : R)AG, (R : CS)AG, (R : CF)AG, (R : R)AG. The language generated by the above grammars are respectively called as (CF : CS)AL, (CF : CF)AL, (CF : R)AL, (R : CS)AL, (R : CF)AL, (R : R)AL.

Definition 9. [3] A non-deterministic (deterministic) two-dimensional online tessellation automaton, referred as 2OTA (2-DOTA) is defined as $\mathcal{A} = (\Sigma, Q, \delta, q_0, F)$ where Σ is the input alphabet, Q is a finite set of states, $q_0 \in Q$ is the initial state, $F \subseteq Q$ is the set of final or accepting states. $\delta : Q \times Q \times \Sigma \rightarrow 2^Q$ ($\delta : Q \times Q \times \Sigma \rightarrow Q$) is the transition function.

A run of \mathcal{A} on a picture $p \in \Sigma^{**}$ consists of associating a state to each position (i, j) of p . Such state is given by the transition function δ and depends on the states already associated to the positions $(i - 1, j)$, $(i, j - 1)$ and on the symbol $p(i, j)$.

At time $t = 0$, an initial state q_0 is associated to all the positions in the border of (\hat{p}) . The computation consists of $\ell_1(p) + \ell_2(p) - 1$ steps. It starts at time $t = 1$ by reading $p(1, 1)$ and associating the state $\delta(q_0, q_0, p(1, 1))$ to position $(1, 1)$. At time $t = 2$ states are simultaneously associated to positions $(1, 2)$ and $(2, 1)$, and so on, to the next diagonals. At time $t = k$, states are simultaneously associated to each positions (i, j) such that $i + j - 1 = k$. A 2OTA \mathcal{A} recognizes a picture p if there exists a run of \mathcal{A} on p such that the state associated to position $(\ell_1(p), \ell_2(p))$ is a final state.

The language accepted by \mathcal{A} is $L(\mathcal{A}) = \{p \in \Sigma^{**} / \text{there is a run of } \mathcal{A} \text{ on } p \text{ such that the state corresponding to the cell } p(m, n) \in F \text{ and } p \text{ is an array of size } (m, n)\}$.

Definition 10. [6] A finite state matrix automaton (FMA) is a 9-tuple $\mathcal{A} = (Q, \Sigma, \Gamma, \delta, \delta', S, F, F', \$)$ where $Q = \bar{Q} \cup Q_1 \cup \dots \cup Q_k$, $Q_i \cap Q_j = \phi$, $i \neq j$, is a finite set of states, Σ is a finite set of input symbols, Γ is a finite

set of stack symbols, $\#(\Gamma) = k$ and each stack symbol corresponds to one and only one Q_i . Each Q_i has an initial state q_i and a final state f_i , $i = 1, \dots, k$. $S = \bigcup_{i=1}^k \{q_i\}$ is the set of start states and $F' = \bigcup_{i=1}^k \{f_i\}$ is the set of transition states. \bar{Q} has an initial state q_0 and $F \subseteq \bar{Q}$ is the set of final states. $\$ \notin \Sigma$ is the end marker. δ is a mapping from $Q_i \times \Sigma$ into the finite subsets of $Q_i \times \{\epsilon\}$, $i = 1, \dots, k$ and from $f_i \times \$$ into the finite subsets of $(S \cup \{q_0\}) \times s_i$ where s_i is the stack symbol corresponding to Q_i , $i = 1, \dots, k$, δ' is the mapping from $\bar{Q} \times \Gamma$ into the finite subsets of \bar{Q} .

The movement of the automaton is as follows: First the input matrix is placed with endmarkers.

$$\begin{matrix} \$ & \dots & \$ \\ a_{m1} & \dots & a_{mn} \\ \dots & \dots & \dots \\ a_{11} & \dots & a_{1n} \end{matrix}$$

The automaton first reads the first column from bottom to top according to δ and when it reaches the first $\$$ it prints a symbol from Γ on the storage (only one storage) and goes to the bottom of the second column and starts from the second column. Similarly it acts on the other columns. If the input matrix is of size $m \times n$ at the end of the δ move, when the input pointer has read the $(m, n)^{th}$ element, the storage will contain n symbols. The automaton then acts on the storage. It starts moving from left to right according to δ' and when it reaches the last letter and after reading it, it must reach a final state; otherwise the automaton rejects the matrix. If it reaches a final state at the end of δ' move, it accepts the input. If the automaton encounters a blank in the middle it stops, rejecting the input.

A configuration is a 4-tuple $(q, (i, j), y, r)$ where q is the current state, (i, j) denotes the position of the input pointer, y is the string of the storage tape and r is the number of cells from the left end of the position of the storage pointer. If $(p, (i, j), y, j)$ is a configuration, ' a ' the $(i, j)^{th}$ element and (p', ϵ) is in $\delta(p, a)$, then $(p, (i, j), y, j) \vdash_{\delta} (p', (i + 1, j), y, j)$, $i = 1, \dots, m$. But if $\$$ is the $(i, j)^{th}$ element and (p', z) is in $\delta(p, \$)$, p in F' , p' in S then $(p, (m + 1, j), y, j) \vdash_{\delta} (p', (1, j + 1), yz, j + 1)$ for all $j \leq n - 1$ but if $j = n$ and $p' = q_0$, then $(p, (m + 1, n), y, n) \vdash_{\delta} (q_0, (m + 1, n), yz, 1)$.

If $(p, (m + 1, n), y, r)$ is a configuration and $\delta'(p, z)$ contains p' then $(p, (m + 1, n), y, r) \vdash_{\delta'} (p', (m + 1, n), y, r + 1)$. \vdash^* is the transitive closure of \vdash .

The set of matrices accepted by the FMA is defined to be

$$\begin{aligned} \mathcal{L}(A) = \{ & [a_{ij}], i = 1, 2, \dots, m; j = 1, 2, \dots, n; m, n \geq 1/a_{ij} \in \Sigma, \\ & (q, (1, 1), \epsilon, 1) \vdash_{\delta}^* (q_0, (m + 1, n), y, 1) \vdash_{\delta'}^* (q', (m + 1, n), y, n) \text{ with } q \text{ in } S, \\ & q' \text{ in } F \text{ and } y \text{ in } \Gamma^+ \}. \end{aligned}$$

Definition 11. [6] A pushdown matrix automaton is a 11-tuple $\mathcal{A} = (\Sigma, \Gamma_1, \Gamma_2, Q, \delta, \delta', S, F, F', Z_0, \$)$, where Q is a finite set of states $Q = \bar{Q} \cup Q_1 \cup \dots \cup Q_k$,

$Q_i \cap Q_j = \phi$ if $i \neq j$. Each Q_i has an initial state q_i and a final state f_i .
 $F' = \bigcup_{i=1}^k \{f_i\}$, $S = \bigcup_{i=1}^k \{q_i\}$, \overline{Q} has an initial state q_0 . $F \subseteq \overline{Q}$ is the set of final states. Γ_1 is the finite set of storage symbols. $\#(\Gamma_1) = k$ and each member of Γ_1 corresponds to one and only one Q_i . Γ_2 is the finite set of second storage symbols. $Z_0 \in \Gamma_2$ is the initial symbol of the second storage. $\$$ is not in Σ and is the end marker. S is the set of start states. F' is the set of transition states. δ is the mapping from $Q_i \times \Sigma$ into finite subsets of $Q_i \times \{\epsilon\}$, $i = 1, \dots, k$ and from $f_i \times \$$ into the finite subsets of $(S \cup \{q_0\}) \times s_i$, $i = 1, 2, \dots, k$ where $s_i \in \Gamma_1$ corresponds to Q_i . δ' is the mapping from $Q \times (\Gamma_1 \cup \{\epsilon\}) \times \Gamma_2$ into finite subsets of $Q \times \Gamma_2^*$. Other things are similar to FMA.

The movement of the automaton is as follows. First the automaton reads each column from the bottom to the top and prints a symbol on the first storage after reading each column. Second it takes this string as the input string and uses the second storage as a pushdown store and sees whether this string is accepted by δ' . If so the automaton enters the final state after reading the string in the first storage. Otherwise it rejects. If the automaton encounters a blank in the middle it stops, rejecting the input.

3 Shuffle on Trajectories

In this section we define an array grammar with shuffle on trajectories and obtain some interesting results. We compare this model with the other existing models for their generative power. We provide results concerning shuffle on trajectories relating to matrix languages.

Definition 12. We define a column shuffle array language over Σ as the set

$$C = \left\{ \left[\begin{array}{c} u_1 \\ u_2 \\ \vdots \\ u_m \end{array} \right] / r, k \geq 1, u_i \in \Sigma^{r \times k}, 1 \leq i \leq m \right\}.$$

Similarly, a row shuffle array language over Σ as the set

$$R = \{ [u_1 \ u_2 \ \dots \ u_m] / c, k \geq 1, u_i \in \Sigma^{k \times c}, 1 \leq i \leq m \}.$$

Definition 13. An array grammar with shuffle on trajectories (AGST) is a construct $G = (\Sigma, B, C, R, T_C, T_R)$ where Σ is an alphabet, B is a finite subset of Σ^{**} called the base of G , C and R are called column and row shuffle array languages over Σ respectively. T_C and T_R are sets of trajectories over the column and row shuffle arrays of C and R respectively.

The direct derivation with respect to G is a binary relation \Rightarrow_{\sqcup_T} on Σ^{**} and is defined as $X \Rightarrow_{\sqcup_T} Y$, where $X, Y \in \Sigma^{**}$ if and only if $Y = X \sqcup_{T_C} U$ or $Y = X \sqcup_{T_R} U$ where $U \in R \cup C$. $\Rightarrow_{\sqcup_T}^*$ is the reflexive transitive closure of \Rightarrow_{\sqcup_T} .

The language generated by G , denoted as $L(G)$ is defined as follows:

$$L(G) = \{Y \in \Sigma^{**} / \exists X \in B \text{ such that } X \Rightarrow^*_{\sqcup_T} Y\}.$$

The family of all languages generated by $AGST$ is denoted by $AGST$.

$AGST_{REG}$, $AGST_{CF}$, $AGST_{CS}$ denote the family of all languages generated by $AGST$ grammars with regular, context-free and context-sensitive languages of trajectories respectively.

Example 2. Let $\Sigma = \{X, \cdot\}$. Let L be the language of L -tokens. L is generated by the following $AGST_{REG}$.

$$G = (\Sigma, B, C, R, T_C, T_R) \text{ where } B = \left\{ \begin{matrix} X & \cdot \\ X & X \end{matrix} \right\},$$

$$R = \{X(\cdot)^n / n \geq 0\}, C = \{X^n / n \geq 1\},$$

$$T_R = \{\ell^n d / n \geq 2\}, T_C = \{r^n u / n \geq 2\}.$$

The derivation is shown below:

$$\begin{array}{cccccccc} X & \cdot & \cdot & \cdot & X & \cdot & \cdot & \cdot \\ X & X & \xrightarrow[\ell^2 d]{\sqcup_{T_R}} & X & \cdot & \xrightarrow[r^2 u]{\sqcup_{T_C}} & X & \cdot & \cdot & \cdot & \xrightarrow[r^3 u]{\sqcup_{T_C}} & X & \cdot & \cdot & \cdot & \xrightarrow[\ell^3 d]{\sqcup_{T_R}} & X & \cdot & \cdot & \cdot \\ & & & X & X & & X & X & X & & X & X & X & X & & X & X & X & X \end{array}$$

Example 3. Let $\Sigma = \{X, \cdot\}$, L the language which includes staircases of X 's. L is generated by the following $AGST_{REG}$, $G = (\Sigma, B, C, R, T_C, T_R)$ where

$$B = \left\{ \begin{matrix} \cdot & \cdot & \cdot & X \\ \cdot & \cdot & \cdot & X \\ X & X & X & X \end{matrix} \right\}, R = \left\{ \left(\begin{matrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{matrix} \right)^n / n \geq 1 \right\},$$

$$C = \left\{ \begin{matrix} \left(\begin{matrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{matrix} \right)_n \\ \left(\begin{matrix} \cdot & \cdot & X \\ \cdot & \cdot & X \\ X & X & X \end{matrix} \right) / m, n \geq 0 \\ \left(\begin{matrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{matrix} \right)_m \end{matrix} \right\}, T_R = \{\ell^m d^n / m, n \geq 1\},$$

$$T_C = \{r^i u^j / i, j \geq 1\}.$$

The sample derivation is given below

$$\begin{array}{cccccccccccc} \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & X & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & X \\ \cdot & \cdot & \cdot & X & \xrightarrow[\ell^3 d^4]{\sqcup_{T_R}} & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & X \\ X & X & X & X & & \cdot & \cdot & X & \xrightarrow[r^4 u^3]{\sqcup_{T_C}} & \cdot & \cdot & \cdot & X & X & X & X & \xrightarrow[r^7 u^3]{\sqcup_{T_C}} & \cdot & \cdot & \cdot \\ & & & & & \cdot & \cdot & X & & \cdot & \cdot & \cdot & X & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ & & & & & X & X & X & X & & X & X & X & X & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{array}$$

```

. . . . . X
. . . . . X
. . . . . X X X X
. . . . . X . . .
. . . X X X X . . .
. . . X . . . . .
X X X X . . . . .
    
```

Staircase

Theorem 1. *The $AGST_{REG}$ intersects $(R : R)AG$.*

Proof. The L token of all sizes of a fixed proportion is generated by $(R : R)AG$, given by $G = (V, I, P, S)$ where $V = V_1 \cup V_2$, $V_1 = \{S\}$, $V_2 = \{A, B\}$, $I = \{., X\}$ and $P = P_1 \cup P_2$ generates L of token of all sizes, the ratio between the two arms of L is 1. $P_1 = \{S \rightarrow (S \ominus A) \oplus B\}$, $L_A = \{X(\cdot)^n/n \geq 0\}$, $L_B = \{(\cdot)^n/n \geq 1\}$

and $P_2 = \left\{ S \rightarrow \begin{matrix} X & \cdot \\ X & X \end{matrix} \right\}$.

The grammar generates L of all sizes, the ratio between the two arms of L is 1. The derivation is shown below:

$$\begin{matrix} X & \cdot \\ X & X \end{matrix} \Rightarrow \begin{matrix} X & \cdot & \cdot \\ X & X & X \end{matrix} \Rightarrow \begin{matrix} X & \cdot & \cdot & \cdot \\ X & \cdot & \cdot & \cdot \\ X & \cdot & \cdot & \cdot \\ X & X & X & X \end{matrix} \Rightarrow \begin{matrix} X & \cdot & \cdot & \cdot \\ X & \cdot & \cdot & \cdot \\ X & \cdot & \cdot & \cdot \\ X & X & X & X \end{matrix}$$

The language is also generated by $AGST_{REG}$ (Example 2). Hence the proof. \square

Let L_1 and L_2 be two array languages and T be a set of trajectories. Then $L_1 \sqcup_T L_2 = \{\alpha \sqcup_t \beta | \alpha \in L_1, \beta \in L_2, t \in T\}$.

We now give an example to illustrate the notion of shuffle of two languages on a set of trajectories.

Example 4. Right-angled isosceles triangles of X 's is generated by the following $(R : R)AG$. Let $G_1 = (V, I, P, S)$ where $V = V_1 \cup V_2$, $V_1 = \{S_1\}$, $V_2 = \{A_1, B_1\}$, $I = \{X, \cdot\}$, $P_1^1 = \{S_1 \rightarrow (S_1 \ominus A_1) \oplus B_1\}$, P_2^1 is the set of intermediate rules given by L_{A_1} and L_{B_1} , $P_3^1 = \left\{ S_1 \rightarrow \begin{matrix} \cdot & X \\ X & X \end{matrix} \right\}$. $L_{A_1} = \{(\cdot)^n/n > 1\}$, $L_{B_1} = \{(X)_n/n \geq 1\}$.

The language $L_1 = L(G_1)$ generated is given below

$$L_1 = \left\{ \begin{matrix} \cdot & \cdot & \cdot & X & \cdot & \cdot & \cdot & X \\ \cdot & X & \cdot & \cdot & X & \cdot & \cdot & X & X \\ X & X & X & \cdot & X & X & X & \cdot & \cdot & \cdot \\ & & & X & X & X & X & & & \end{matrix} , \dots \right\}$$

Consider another language L_2 , consisting of token L of all sizes and of all proportions generated by the following $(R : R)AG$. Let $G_2 = (V, I, P, S)$ where

$V = V_1 \cup V_2$, $V_1 = \{S\}$, $V_2 = \{A_2, B_2\}$, $I = \{X, \cdot\}$, $P_1^2 = \{S_2 \rightarrow (S_2 \ominus A_2) \oplus B_2\}$, P_2^2 is the set of intermediate rules given by L_{A_2} and L_{B_2} , $P_3^2 = \left\{ S_2 \rightarrow \begin{matrix} X & \cdot \\ X & X \end{matrix} \right\}$.

$$L_{A_2} = \{X (\cdot)^n / n \geq 0\}, L_{B_2} = \left\{ \begin{matrix} (\cdot)^n \\ X \end{matrix} / n \geq 1 \right\}.$$

The language thus generated is

$$L_2 = L(G_2) = \left\{ \begin{matrix} X & \cdot & \cdot \\ X & X & \cdot \\ X & X & X \end{matrix}, \begin{matrix} X & \cdot & \cdot & \cdot \\ X & X & X & X \end{matrix}, \dots \right\},$$

$$L_1 \sqcup_{T_C} L_2 = \left\{ \begin{matrix} \cdot & X & X & \cdot & \cdot & X & \cdot & \cdot & X & \cdot \\ X & X & X & X & X & X & X & X & X & X \end{matrix}, \dots \right\}, T_C = \{(ru)^i, i \geq 1\}$$

$$L_1 \sqcup_{T_R} L_2 = \left\{ \begin{matrix} X & \cdot & \cdot \\ X & \cdot & \cdot & \cdot & X \\ \cdot & X & X & \cdot & \cdot \\ X & X & \cdot & X & X \\ X & X & X & X & X \\ X & X & X \end{matrix} \right\}, T_R = \{(\ell d)^j, j \geq 1\}$$

$L_1 \sqcup_{T_C} L_2$, $L_1 \sqcup_{T_R} L_2$ are $(R : R)AL$, and are generated by G_3 and G_4 respectively, where $G_3 = (V, I, P, S_3)$, $V = V_1 \cup V_2$, $V_1 = \{S_3\}$, $V_2 = \{A_3, B_3\}$, $I = \{X, \cdot\}$,

$P_1^3 = \{S_3 \rightarrow (S_3 \ominus A_3) \oplus B_3\}$, P_2^3 is the set of intermediate rules given by L_{A_3} and L_{B_3} , $P_3^3 = \left\{ S_3 \rightarrow \begin{matrix} \cdot & X & X & \cdot \\ X & X & X & X \end{matrix} \right\}$. $L_{A_3} = \{\cdot X (\cdot)^n / n \geq 1\}$,

$$L_{B_3} = \left\{ \begin{matrix} (X & \cdot)^n \\ X & X \end{matrix} / n \geq 1 \right\} \text{ and } G_4 = (V, I, P, S_4), V = V_1 \cup V_2, V_1 = \{S_4\},$$

$V_2 = \{A_4, B_4\}$, $I = \{X, \cdot\}$, $P_1^4 = \{S_4 \rightarrow (S_4 \ominus A_4) \oplus B_4\}$,

P_2^4 is the set of intermediate rules given by L_{A_4} and L_{B_4} ,

$$P_3^4 = \left\{ S_4 \rightarrow \begin{matrix} X & \cdot \\ \cdot & X \\ X & X \\ X & X \end{matrix} \right\}, L_{A_4} = \left\{ X \cdot \left(\begin{matrix} \cdot \\ \cdot \end{matrix} \right)^n / n \geq 0 \right\}.$$

$$L_{B_4} = \left\{ \begin{matrix} \left(\begin{matrix} \cdot \\ X \end{matrix} \right)_n \\ \cdot \\ X \\ \cdot \\ X \\ X \\ X \end{matrix} / n \geq 0 \right\}.$$

The following theorem is with respect to RMLs and a regular language.

Theorem 2. *Let $T \subseteq \{r, u\}^* \cup \{\ell, d\}^*$ be a set of trajectories. If T is a regular language and L_1, L_2 are all regular matrix languages (RML), then $L_1 \sqcup_T L_2$ is a regular matrix language.*

Proof. Let T be a regular language. L_1, L_2 are two regular matrix languages over the same alphabet Σ . Let $\mathcal{A}_1 = (\Sigma, \Gamma_1, Q_1, \delta_1, \delta'_1, S_1, F_1, F'_1, \$)$ and $\mathcal{A}_2 = (\Sigma, \Gamma_2, Q_2, \delta_2, \delta'_2, S_2, F_2, F'_2, \$)$ are finite state matrix automata such that $L(\mathcal{A}_1) = L_1$ and $L(\mathcal{A}_2) = L_2$, where $Q_1 = \overline{K_1} \cup K_{11} \cup K_{12} \cup \dots \cup K_{1k}$, $K_{1i} \neq K_{1j}$ for $i \neq j$ is a finite set of states and $Q_2 = \overline{K_2} \cup K_{21} \cup K_{22} \cup \dots \cup K_{2k}$, where Σ is a finite set of input symbols. Γ_1 and Γ_2 are stack symbols correspond to one and only one K_{1i} and K_{2i} respectively. Each K_{1i} and K_{2i} has an initial state q_{1i} and q_{2i} . The final states are f_{1i} and f_{2i} for $i = 1, 2, \dots, k$. $S_1 = \bigcup_{i=1}^k \{q_{1i}\}$ and $S_2 = \bigcup_{i=1}^k \{q_{2i}\}$ are the sets of start states. $F'_1 = \bigcup_{i=1}^k \{f_{1i}\}$ and $F'_2 = \bigcup_{i=1}^k \{f_{2i}\}$ are the sets of transition states. $\overline{K_1}$ and $\overline{K_2}$ have initial state q_{10} and q_{20} respectively. $F_1 \subseteq \overline{K_1}$, $F_2 \subseteq \overline{K_2}$ are the set of final states. $\$ \notin \Sigma$ is the endmarker, δ_1 is a mapping from $K_{1i} \times \Sigma$ into the finite subsets of $K_{1i} \times \{\xi\}$, $i = 1 \dots k$ and from $f_{1i} \times \$$ into the finite subsets of $(S_1 \cup \{q_{10}\}) \times S_{1i}$ where S_{1i} is the stack symbol corresponding to K_{1i} , similarly δ_2 can be defined. δ'_1 is the mapping from $\overline{K_1} \times \Gamma$ into the finite subsets of $\overline{K_1}$. Similarly we define δ'_2 . Let $\mathcal{A}_T = (\{r, u, \ell, d\}, Q_T, \delta_T, q_0^T, F_T)$ be a finite deterministic automaton such that $L(\mathcal{A}_T) = T$.

We define a finite state matrix automaton as a 9-tuple $\mathcal{A} = (\Sigma, \Gamma, Q, \delta, \delta', S, F, F', \$)$ such that $L(\mathcal{A}) = L_1 \sqcup_T L_2$, where Q is a finite set of states, $Q = \overline{R} \cup R_1 \cup \dots \cup R_k$, $R_i \cap R_j = \emptyset$, if $i \neq j$. Each R_i has an initial state q_i and a final state f_i , $i = 1 \dots k$. $S = \bigcup_{i=1}^k \{q_i\}$, \overline{R} has an initial state q_0 , $F \subseteq \overline{R}$ is the set of final states. $\$ \notin \Sigma$ is the end marker. Formally $Q = Q_1 \times Q_T \times Q_2$, $\overline{R} = \{(q_0^1, q_0^T, q_0^2)\}$, $F = F_1 \times F_T \times F_2$ and $F' = F'_1 \times F'_2$. Informally \mathcal{A} on an input $p \in \Sigma^{**}$ simulates non deterministically \mathcal{A}_1 or \mathcal{A}_2 and from time to time changes the simulation from \mathcal{A}_1 to \mathcal{A}_2 or from \mathcal{A}_2 to \mathcal{A}_1 . Each change determines a transition in \mathcal{A}_T as follows: a change from \mathcal{A}_1 to \mathcal{A}_2 is interpreted as u or d and a change from \mathcal{A}_2 to \mathcal{A}_1 is interpreted as r or ℓ respectively. The input p is accepted by \mathcal{A} if and only if each of $\mathcal{A}_1, \mathcal{A}_2$ and \mathcal{A}_T accepts L_1, L_2 and T respectively.

The transition δ is defined as

$$\delta((q_1, q_T, q_2), a) = \{(\delta_1(q_1, a), \delta_T(q_T, r), q_2), (q_1, \delta_T(q_T, u), \delta_2(q_2, a)), (\delta_1(q_1, a), \delta_T(q_T, \ell), q_2), (q_1, \delta_T(q_T, d), \delta_2(q_2, a))\}$$

where $q_1 \in Q_1$, $q_2 \in Q_2$, $a \in \Sigma$, $q_T \in Q_T$. δ' is a mapping from $\overline{R} \times \Gamma$ into the finite subsets of \overline{R} .

We can easily verify that $L(\mathcal{A}) = L_1 \sqcup_T L_2$ and hence $L_1 \sqcup_T L_2$ is a regular matrix language. □

The next theorem is with respect to RMLs and a CFL.

Theorem 3. *Let $T \subseteq \{r, u\}^* \cup \{\ell, d\}^*$ be a set of trajectories. If T is a context-free language and L_1, L_2 are all regular matrix languages (RML), then $L_1 \sqcup_T L_2$ is a context-free matrix language.*

Proof. Let T the set of trajectories be a context-free language. Consider two regular matrix array languages L_1, L_2 . Without loss of generality, we may assume that L_1 and L_2 are over the same alphabet. Let $\mathcal{A}_i = (\Sigma, \Gamma_i, Q_i, \delta_i, \delta'_i, S_i, F_i, F'_i, \$)$ be a finite state matrix automata such that $L(\mathcal{A}_i) = L_i$, for $i = 1, 2$. Let $\mathcal{A}_p = (\{r, u, \ell, d\}, \Gamma_T, Q_T, q_0^T, Z_0^T, F_T, \delta_T)$ be the pushdown automaton such that $L(\mathcal{A}_p) = T$, where Q_T is the finite set of states, Γ_T is the stack alphabet, $q_0^T \in Q_T$ is the initial state, $Z_0^T \in \Gamma_T$ is the initial stack symbol, $F_T \subseteq Q_T$ is the set of final states and δ_T is the transition mapping defined as $\delta_T : Q_T \times (\{r, u, \ell, d\} \cup \{\lambda\}) \times \Gamma_T \rightarrow 2^{Q_T \times \Gamma_T}$.

We construct a pushdown matrix automaton $\mathcal{A} = (\Sigma, \Gamma_1, \Gamma_2, Q, \delta, \delta', S, F, F', Z_0, \$)$ such that $L(\mathcal{A}) = L_1 \sqcup_T L_2$, where Q is a finite set of states $Q = \overline{Q} \cup Q_1 \cup \dots \cup Q_k$, $Q_i \cap Q_j = \emptyset$ if $i \neq j$. Each Q_i has an initial state q_i and a final state f_i . $F' = \bigcup_{i=1}^k \{f_i\}$, $S = \bigcup_{i=1}^k \{q_i\}$, \overline{Q} has an initial state q_0 . $F \subseteq \overline{Q}$ is the set of final states. Γ_1 is the finite set of storage symbols. $\#(\Gamma_1) = k$ and each member of Γ_1 corresponds to one and only one Q_i . Γ_2 is the finite set of second storage symbols. $Z_0 \in \Gamma_2$ is the initial symbol of the second storage. $\$$ is not in Σ and is the end marker. Informally \mathcal{A} on an input $p \in \Sigma^{**}$ simulates non-deterministically \mathcal{A}_1 or \mathcal{A}_2 and from time to time changes the simulation from \mathcal{A}_1 to \mathcal{A}_2 or from \mathcal{A}_2 to \mathcal{A}_1 . Each change determines a transition in \mathcal{A}_T as follows:

A change from \mathcal{A}_1 to \mathcal{A}_2 is interpreted as u or d and a change from \mathcal{A}_2 to \mathcal{A}_1 is interpreted as r or ℓ respectively. The input p is accepted by \mathcal{A} if and only if each of $\mathcal{A}_1, \mathcal{A}_2$ and \mathcal{A}_p accepts L_1, L_2 and L_T respectively.

Formally $Q = Q_1 \times Q_T \times Q_2$. $\overline{Q} = \{(q_0^1, q_0^T, q_0^2)\}$, $F = F_1 \times F_T \times F_2$ and $F' = F'_1 \times F'_2$. The transition function δ is defined as $\delta : Q \times Q \times \Sigma \rightarrow 2^Q$.

The transition δ is given as

$$\delta((q_1^1, q_T, q_1^2), (q_2^1, q_T, q_2^2), a, X) = \bigcup_{(s, \alpha) \in \delta_T(q_T, r, X)} \{((\delta_1(q_1^1, q_2^1, a), s, q_1^2), \alpha)\} \cup \bigcup_{(s', \alpha') \in \delta_T(q_T, u, X)} \{((q_1^1, s', \delta_2(q_1^2, q_2^2, a)), \alpha')\}$$

and

$$\delta((q_1^1, q_T, q_1^2), (q_2^1, q_T, q_2^2), \lambda, X) = \bigcup_{(s, \alpha) \in \delta_T(q_T, \lambda, X)} \{((\delta_1(q_1^1, q_2^1, \lambda), s, q_1^2), \alpha)\}$$

where $q_1^1, q_2^1 \in Q_1, q_T \in Q_T, q_1^2, q_2^2 \in Q_2, X \in \Gamma, \alpha, \alpha' \in \Gamma^*, \lambda \in T$. Additionally,

$$\delta((q_1^1, q_T, q_1^2), (q_2^1, q_T, q_2^2), a, X) = \bigcup_{(s, \alpha) \in \delta_T(q_T, \ell, X)} \{((\delta_1(q_1^1, q_2^1, a), s, q_1^2), \alpha)\} \cup \bigcup_{(s', \alpha') \in \delta_T(q_T, d, X)} \{((q_1^1, s', \delta_2(q_1^2, q_2^2, a)), \alpha')\}$$

and

$$\delta((q_1^1, q_T, q_1^2), (q_2^1, q_T, q_2^2), \lambda, X) = \bigcup_{(s, \alpha) \in \delta_T(q_T, \lambda, X)} \{((\delta_1(q_1^1, q_2^1, \lambda), s, q_1^2), \alpha)\}$$

where $q_1^1, q_1^2 \in Q_1, q_T \in Q_T, q_2^1, q_2^2 \in Q_2, X \in \Gamma, \alpha, \alpha' \in \Gamma^*$. δ' is the mapping from $\overline{Q} \times (\Gamma_1 \cup \{\epsilon\}) \times \Gamma_2$ into finite subsets of $\overline{Q} \times \Gamma_2^*$. Clearly $L_1 \sqcup_T L_2$ is a context-free matrix language. \square

Now we have a result with respect to two recognizable languages L_1 and L_2 and a regular language T .

Theorem 4. *Let $T \subseteq \{r, u\}^* \cup \{\ell, d\}^*$ be a set of trajectories. For all recognizable array languages $L_1, L_2, L_1 \sqcup_T L_2$ is a recognizable array language, if T is regular.*

Proof. Let T be a regular language. L_1, L_2 are two recognizable array languages over the same alphabet Σ . Let $\mathcal{A}_i = (\Sigma, Q_i, \delta_i, q_0^i, F_i)$ be a deterministic two dimensional online tessellation automaton such that $L(\mathcal{A}_i) = L_i$ for $i = 1, 2$. Also $\mathcal{A}_T = (\{r, u, \ell, d\}, Q_T, \delta_T, q_0^T, F_T)$ be a deterministic finite state automaton such that $L(\mathcal{A}_T) = T$.

We define a finite non-deterministic online tessellation automaton $\mathcal{A} = (\Sigma, Q, \delta, Q_0, F)$ such that $L(\mathcal{A}) = L_1 \sqcup_T L_2$. Informally \mathcal{A} on an input $p \in \Sigma^{**}$ simulates non-deterministically \mathcal{A}_1 or \mathcal{A}_2 and from time to time changes the simulation from \mathcal{A}_1 to \mathcal{A}_2 or from \mathcal{A}_2 to \mathcal{A}_1 . Each change determines a transition in \mathcal{A}_T as follows: a change from \mathcal{A}_1 to \mathcal{A}_2 is interpreted as u or d and a change from \mathcal{A}_2 to \mathcal{A}_1 is interpreted as r or ℓ respectively. The input p is accepted by \mathcal{A} if and only if each of $\mathcal{A}_1, \mathcal{A}_2$ and \mathcal{A}_T accepts L_1, L_2 and T respectively.

Formally $Q = Q_1 \times Q_T \times Q_2, Q_0 = \{(q_0^1, q_0^T, q_0^2)\}, F = F_1 \times F_T \times F_2$. The transition δ is defined as follows:

$$\delta : Q \times Q \times \Sigma \rightarrow 2^Q.$$

The transition δ is defined as follows:

$$\begin{aligned} &\delta((q_1^1, q_T, q_1^2), (q_2^1, q_T, q_2^2), a) \\ &= \{(\delta_1(q_1^1, q_2^1, a), \delta_T(q_T, r), q_1^2), (q_1^1, \delta_T(q_T, u), \delta_2(q_1^2, q_2^2, a)), \\ &\quad (\delta_1(q_1^1, q_1^2, a), \delta_T(q_T, \ell), q_1^2), (q_1^1, \delta_T(q_T, d), \delta_2(q_1^2, q_2^2, a))\} \end{aligned}$$

where $q_1^1, q_1^2 \in Q_1, q_T \in Q_T, q_2^1, q_2^2 \in Q_2, a \in \Sigma$. It can be easily verified that $L(\mathcal{A}) = L_1 \sqcup_T L_2$ and hence $L_1 \sqcup_T L_2$ is a recognizable language. \square

4 Conclusion

The use of shuffle operation in the theory of concurrency and parallel composition is well known. This operation is used to yield formal languages. The shuffle on finite and infinite words have been investigated extensively. But the study of the shuffle operation on finite arrays is in the initial stage. So in this paper, we have made an attempt to study in depth the use of shuffle operation on different languages of finite arrays to provide new classes of languages of arrays and images. We have defined an array grammar with shuffle on trajectories over finite arrays and obtained interesting results. Based on the studies in [11,2] the results shown in section 3 can be extended to $\omega\omega$ -array languages.

References

1. Kadrie, A., Rajkumar Dare, V., Thomas, D.G., Subramanian, K.G.: Algebraic properties of the shuffle over omega-trajectories. *Inf. Process. Lett.* 80(3), 139–144 (2001)
2. Dare, V.R., Subramanian, K.G., Thomas, D.G., Siromoney, R.: Infinite arrays and recognizability. *International Journal of Pattern Recognition and Artificial Intelligence* 14(4), 525–536 (2000)
3. Giammarresi, D., Restivo, A.: Two-dimensional languages. In: Salomaa, A., Rozenberg, G. (eds.) *Handbook of Formal Languages*, vol. 3, pp. 215–267. Springer, Heidelberg (1997)
4. Helen Chandra, P., Martin-Vide, C., Subramanian, K.G., Van, D.L., Wang, P.S.P.: Parallel contextual array grammars and trajectories. In: Chen, C.H., Wang, P.S.P. (eds.) *Handbook of Pattern Recognition and Computer Vision*, 3rd edn., pp. 55–70. World Scientific, Singapore (2004)
5. Helen Chandra, P., Subramanian, K.G., Thomas, D.G.: Parallel contextual array grammars and languages. *Electronic Notes in Discrete Mathematics* 14, 537–550 (2000)
6. Siromoney, G., Siromoney, R., Krithivasan, K.: Abstract families of matrices and picture languages. *Computer Graphics and Image Processing. I*, 284–470 (1972)
7. Siromoney, G., Siromoney, R., Krithivasan, K.: Picture Languages with array rewriting rules. *Information and Control* 22, 447–470 (1973)
8. Mateescu, A., Rozenberg, G., Salomaa, A.: Shuffle on trajectories: syntactic constraints. *Theoretical Computer Science* 197, 1–56 (1998)

Planar Configurations Induced by Exact Polyominoes

Daniela Battaglino¹, Andrea Frosini², and Simone Rinaldi¹

¹ Dipartimento di Matematica e Informatica
Università di Siena, Siena, Italy
{battaglino3,rinaldi}@unisi.it

² Dipartimento di Sistemi e Informatica
Università di Firenze, Firenze, Italy
andrea.frosini@unifi.it

Abstract. An unknown planar discrete set of points A can be inspected by means of a probe P of generic shape that moves around it, and reveals, for each position, the number of its elements as a magnifying glass. All the data collected during this process can be naturally arranged in an integer matrix that we call the scan of the starting set A w.r.t. the probe P .

When the probe is a rectangle, a set A whose scan is homogeneous shows a strong periodical behavior, and can be decomposed into smaller homogeneous subsets. Here we extend this result, which has been conjectured true for all the exact polyominoes, to the class of diamonds, and we furnish experimental evidence of the decomposition theorem for exact polyominoes of small dimension, using the mathematical software Sage.

1 Introduction

The aim of *discrete tomography* is the retrieval of geometrical information about a physical structure, regarded as a finite set of points, i.e. its primary constituents, from measurements, generically known as *projections*, of the number of the structure's constituents that lie on (discrete) lines with fixed scopes. A common simplification is to represent a finite physical structure A as a binary matrix, where an entry is 1 or 0 according to the presence or the absence of material at the corresponding point of the lattice (see [6] for a survey).

Following [7], here we consider a model where data from a structure are collected not with respect to lines of fixed scope, but using of a probe P , much as we might examine a specimen under a microscope or magnifying glass. For each possible position of the probe, we count the number of visible points, or equivalently, in the binary matrix model, the number of the 1s inside the probe.

The collected data can be naturally arranged as an integer matrix, say the *scan* of A with respect to the probe P , whose values range from 0, when no 1s lie inside P , to the dimension of the probe itself, regarded as the maximum number of elements of the matrix that may lie inside it.

In [7], planar binary configurations are considered, i.e. infinite discrete sets, setting aside the problems concerning border phenomena that may eventually arise in finite configurations; such a simplification is possible thanks to the very local behavior of the notion of scan. These infinite sets are inspected by means of rectangular probes called windows due to their shapes: if the obtained scans have a constant value, i.e. they are *homogeneous*, then the related sets show strong periodical behaviors. Furthermore, in this setting, a decomposition theorem, that allows to split each set having homogeneous scan into minimal homogeneous subsets, holds.

The two highlighted results lead to the definition of a reconstruction strategy for finite discrete sets from rectangular scans obtained in [4] and [5].

The present work constitutes a step towards the generalization of the decomposition theorem given in [7] to exact probes i.e. those connected sets that tile the plane by translation: an expected result since that very first paper.

So, our studies start by considering homogeneous scans obtained from a class of exact probes called diamonds, and we give some properties of the related discrete sets, again including the periodicity of their elements; using these results we hit the goal of a new decomposition theorem.

Finally, we move to generic exact probes of small dimension, and we give experimental evidence of the possible decompositions of infinite sets having homogeneous scans into minimal homogeneous subsets, by using the mathematical software system **Sage**.

2 Definitions and Results

Borrowing the notation from [7], let us consider a set of points A in the lattice \mathbb{Z}^2 and let U_A be its characteristic function, i.e. for all $z \in \mathbb{Z}^2$

$$\begin{aligned} U_A(z) &= 1 \text{ if } z \in A \\ U_A(z) &= 0 \text{ if } z \notin A. \end{aligned}$$

Let us define the *probe* $P = \{p_0 = (0, 0), p_1, p_2, \dots, p_n\}$ to be a finite set of points of \mathbb{Z}^2 including the origin, and that we will use to collect information about the set A , as a sort of window that moves along it, and reveals, position by position, the total number of elements of A that lie inside P . The A set is represented using a binary matrix, where every point of A corresponds to an (i, j) position which is set to 1, while the probe P is represented through a union of unitary cells. The center of these cells corresponds to a point of P , (see Fig 1).

The collected data can be arranged in a two dimensional infinite array P_A called the *scan* of A with respect to P , briefly P -scan of A , and whose generic element is defined as follows:

$$P_A(i, j) = \text{card}\{p \in (i, j) + P : U_A(p) = 1\},$$

where $(i, j) + P$ means the translation of the probe P by the vector (i, j) (see Fig 2 (a) and (b)).

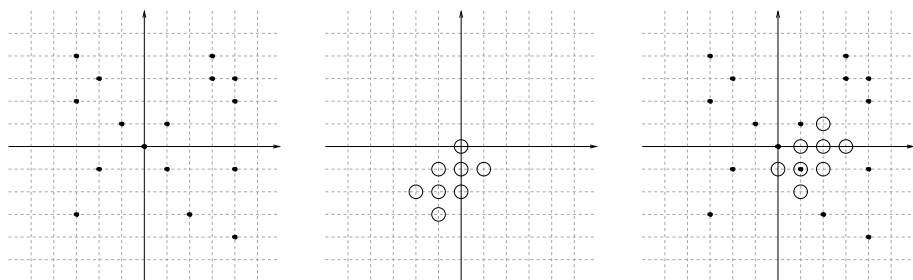


Fig. 1. (a): a finite set of points of \mathbb{Z}^2 , A ; (b): (the contour of) a finite set of points used as a probe, P ; (c): some elements of the infinite scan related to the set in (a) and the probe in (b) translated by the vector $(2, 1)$

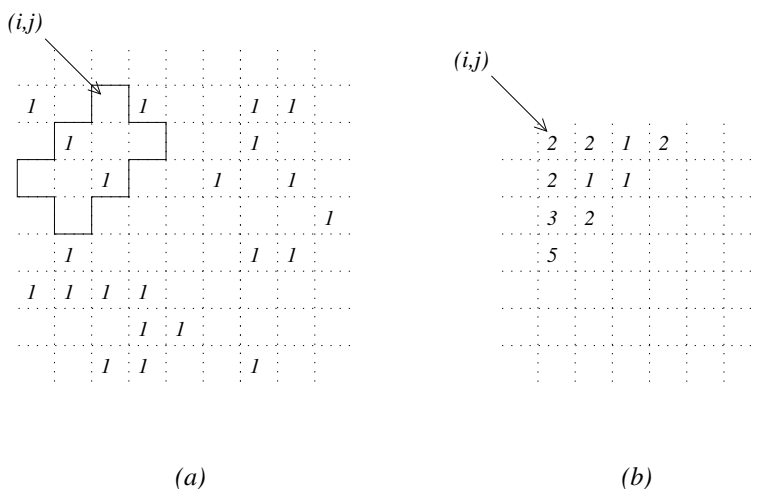


Fig. 2. (a): an infinite binary matrix representing a discrete set of points. The highlighted probe is translated by the vector (i, j) ; (b): some elements of the infinite scan related to the set and the probe in (a).

The set A is *homogeneous* of degree k , say k -homogeneous, with respect to P if all the elements of the scan P_A have constant value k , that is for every translations of P over \mathbb{Z}^2 the same number k of points of A appears in the probe.

A set A constitutes a coverage of the plane with respect to the probe P if and only if

$$\mathbb{Z}^2 = A + P,$$

where $+$ stand for the Minkowski sum, i.e. for each $z \in \mathbb{Z}^2$ there exist one element $a \in A$ and one element $p \in P$ such that $z = a + p$.

If the Minkowski sum is non ambiguous, i.e. each element $z \in \mathbb{Z}^2$ can be obtained in only one way from the elements of A and the elements of P , then we say that A is a tiling of the plane, and we indicate the Minkowski operator with \oplus .

Finally, the set A is periodic with respect to the (integer) vector (i, j) if it holds that $a \in A$ if and only if $(i, j) + a \in A$.

The following result, given in [7], strictly relates homogeneous scans with the shape of the used probe:

Theorem 1. *There is a subset of \mathbb{Z}^2 , A , homogeneous of degree 1 for P if and only if the probe P tiles the plane by translation.*

The notion of *exact* set has been defined by Beauquier and Nivat in [1], and it characterizes all those connected sets P , say *polyominoes*, that tile the plane by translation, i.e. for which there exists an infinite set A such that $\mathbb{Z}^2 = P \oplus A$.

In the same paper it has also been proved that each 1-homogeneous set is periodic with respect to one or two directions according to the characteristics of the related tile (see Fig. 3).

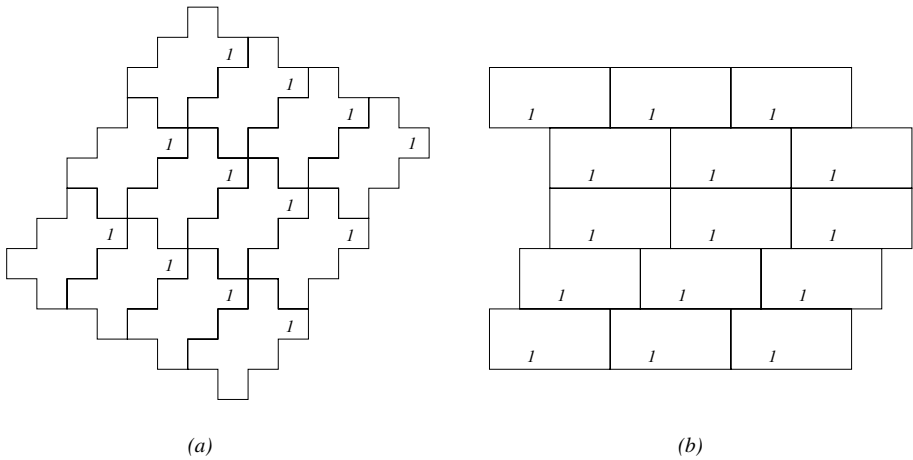


Fig. 3. Different probes that tile the plane by translation: in (a) a 1-homogeneous set having periodicity along the two directions $(2, 3)$ and $(2, -1)$, while in (b) a 1-homogeneous set having periodicity only along the direction $(4, 0)$

Exact sets have been widely studied from different points of view: they can be detected and generated quickly [3], those having convexity properties are enumerated [2], and the number of different tilings of the plane that are related to a given one is conjectured [8].

Furthermore, in [7], Nivat studies the configurations of the plane that are homogeneous with respect to probes with rectangular shape, the simplest subclass of exact polyominoes, and proves a theorem that constitutes the basis of our studies:

Theorem 2. *Let P be a rectangular probe of dimension $p \times q$, with $p, q \in \mathbb{N}$. If the P -scan of a discrete set A is k -homogeneous, then A can be decomposed into k subsets that are 1-homogeneous with respect to P .*

Here, we extend this result to the the class of diamonds polyominoes, simply diamond, and successively we furnish experimental evidence for its full generalization to each exact polyominoes. The technique we use in this paper is different from those in [7].

3 Probing the Plane with Diamond Polyominoes

For each integer $n \geq 1$, we define *diamond* of dimension n , say D_n , the set of points of \mathbb{Z}^2 whose L_1 distance from $(\frac{n}{2}, \frac{n}{2})$ is less than or equal to $\lceil \frac{n}{2} \rceil$.

For each n , we indicate the point $(\frac{n}{2}, \frac{n}{2})$ as *center* of the diamond D_n ; note that, for n odd, the center of D_n has no integer coordinates (see Fig. 4).

The elements D_n , with n odd, are known as *aztec diamonds*, and constitute one of the most interesting and widely studied class of discrete sets in the fields of combinatorics. Since we want to use the diamonds to probe planar configurations of points we have to translate them in order to include the point $(0, 0)$ in their interiors.

The following properties directly lead to the generalization of Theorem 2 to the class of diamonds:

Property 1. Each diamond D_n is an exact polyomino (see Fig. 5 for an example).

Property 2. Let A be a 2-homogeneous set with respect to D_n . For each point $z \in \mathbb{Z}^2$, there exist exactly two points $a_1, a_2 \in A$ such that $z \in (a_1 + D_n) \cap (a_2 + D_n)$ (i.e. we say that A is a 2-coverage of z).

Proof. Let us proceed by contradiction assuming that there exist at least three points $a_1, a_2, a_3 \in A$ such that $z \in (a_1 + D_n) \cap (a_2 + D_n) \cap (a_3 + D_n)$ (if we assume the existence of at most one point of A covering z , the proof is analogous).

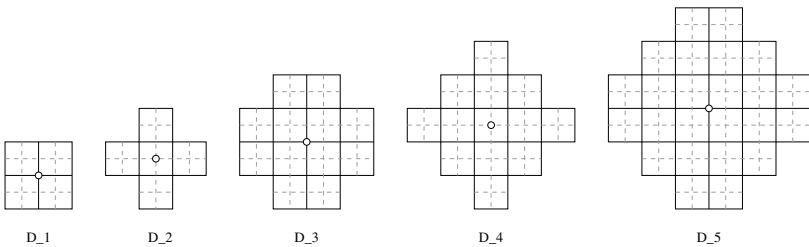


Fig. 4. The first five diamonds polyominoes whose centers are highlighted

Let c be the center of D_n , and, for each $i \in \{1, 2, 3\}$, c_i be the center of $(a_i + D_n)$. Since it holds $a_i + c = c_i$ then

$$\left\lceil \frac{n}{2} \right\rceil \geq L_1(z, c_i) = L_1(z, a_i + c) = L_1(z - c, a_i).$$

So, the translated probe $(z - c) + D_n$ contains the three points a_1 , a_2 , and a_3 against the assumption of the 2-homogeneity of A . □

Property 3. If a discrete set A is a 2-coverage with respect to the diamond D_n in each point $z \in \mathbb{Z}^2$, then A can be decomposed into two sets A_1 and A_2 that are 1-homogeneous with respect to D_n .

The proof is straightforward: we start by randomly choosing a point $a \in A$ and we move it into A_1 . Now, we move in A_1 each point a' that is 2-covered by A and such that $a + D_n$ and $a' + D_n$ have a common edge. Repeating this process we obtain a set A_1 that is a full coverage of \mathbb{Z}^2 , since D_n is exact; what remains in A is the set A_2 . Since this process can be applied up to translations of the probe D_n , then the result follows.

Theorem 3. *Each 2-homogeneous set A with respect to the diamond D_n admits a decomposition into two sets that are 1-homogeneous respect D_n .*

The proof directly follows from Properties [2](#) and [3](#).

Corollary 1. *Each k -homogeneous set with respect to D_n can be decomposed into k sets that are 1-homogeneous.*

We can suppose to proceed in the same way as used in the case of 2-homogeneity, repeating the procedure k times. At every generic step t we have t sets 1-homogeneous and a set $k - t$ -homogeneous.

This last corollary completes the generalization of Theorem [2](#) to the class of diamonds polyominoes. Some further simple properties of k -homogeneous sets can be highlighted:

Property 4. For each diamond D_n , there exist two different sets of points A and A' that are 1-homogeneous with respect to it. If n is even, then the two sets are periodic with respect to the couple of vectors $(\frac{n}{2} + 1, \frac{n}{2})$ and $(\frac{n}{2} + 1, -\frac{n}{2})$ or $(\frac{n}{2}, \frac{n}{2} + 1)$ and $(\frac{n}{2}, -\frac{n}{2} - 1)$, while if n is odd, then they are periodic with respect to the couple of vectors $(\lfloor \frac{n}{2} \rfloor + 1, \lfloor \frac{n}{2} \rfloor)$ and $(\lfloor \frac{n}{2} \rfloor, -\lfloor \frac{n}{2} \rfloor - 1)$ or $(\lfloor \frac{n}{2} \rfloor, \lfloor \frac{n}{2} \rfloor + 1)$ and $(\lfloor \frac{n}{2} \rfloor + 1, -\lfloor \frac{n}{2} \rfloor)$.

Figure [5](#) shows the two different 1-homogeneous sets related to the diamond D_4 and their periodicity.

Property 5. If A is k -homogeneous with respect to D_n , then A is periodic with respect to exactly one couple of vectors listed in Property [4](#) related to the parity of n .

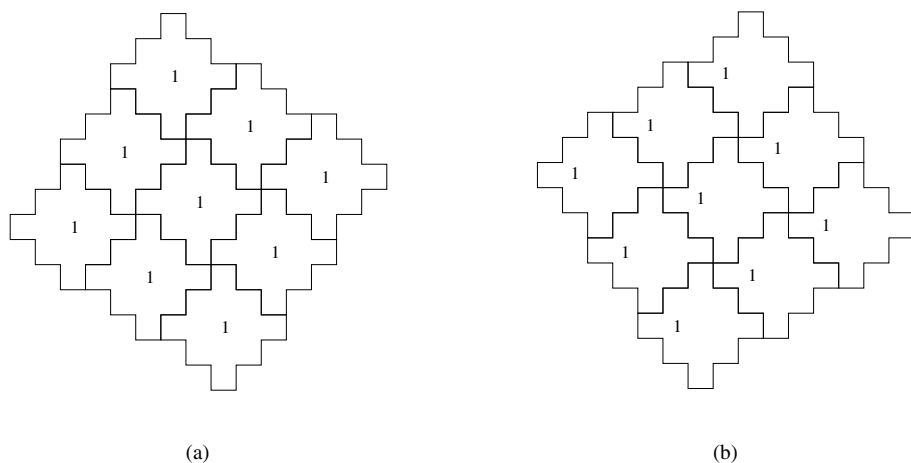


Fig. 5. The two 1-homogeneous sets with respect to D_4 : the elements of (a) have periodicity $(2, 3)$ and $(3, -2)$, while the elements of (b) have periodicity $(3, 2)$ and $(2, -3)$

The proof immediately follows after observing that, for each n , directions of periodicity are coprime coordinate by coordinate.

Further studies will concern the possibility of extending the results in Theorem 3 to a more general class of diamonds, say pseudo-diamonds, whose staircase edges may vary in slope.

Such a steps will proceed in the direction of a final decomposition theorem that will hold for the whole class of exact polyominoes, as expected since the very first approach to the problem in [7].

In the next paragraph, we furnish experimental evidence to this result by a computer program that generates all the exact polyominoes of a given dimension and, for each of them, inspects all its possible k -homogeneous discrete sets, finding a decomposition into 1-homogeneous sets. All the experimental results have been obtained using the mathematical software SAGE.

4 Experimental Results Obtained with Sage

We know that the properties of periodicity hold also for windows with a shape of a pseudo-square or a pseudo-hexagon. Thus, it is natural to check, in an experimental way, the validity of the theorem of decomposition for these types of windows.

So, we wrote a program which takes as input all pseudo-square or pseudo-hexagon polyominoes of dimension n and an integer k ($k \leq (\dim P)/2$), and returns all the $\{0, 1\}$ -configurations which are k -homogeneous with respect to the considered polyomino.

To write our program we used **Sage**, which is a free open-source mathematics software system licensed under the GPL. It combines the power of many existing open-source packages into a common Python-based interface. The reader can find all useful information concerning **Sage** on the web site <http://www.sagemath.org>.

Since we can consider all possible configurations as the sum of 1-homogeneous configurations and since these are periodic with period determined by the examined polyomino, we have only a fixed number of final configurations. These configurations are exactly the ones obtained from the combinations of 1-homogeneous configurations.

Then, if we consider the theorem valid with this new set of windows, the program output should return a few possible configurations accordingly to the periodicity induced by the tiling of the plane by the polyomino and from the degree of homogeneity.

The general idea of the algorithm is based on the fact that the theorem can be not true. In this case the number of possible configurations can be very high.

We posted the work calculating step by step all possible positions combinations, in which we can insert the value 1. For every possible resulting combination the described procedure will then be recursively applied.

Our work can be divided in three steps:

step 1. At first, we generate a list of pseudo-squares or pseudo-hexagons of fixed semi-perimeter n , with $n \leq 12$.

step 2. After that, every generated polyomino P is represented in the center of a matrix of dimensions $[p] \times [q]$, where p and q are the triple of the length of the basis and of the height of the minimal bounding rectangle of P , respectively.

At this point P is translated, by one step, following a spiral direction in the eight directions surrounding it. This operation is repeated for every new obtained copy of P , deleting the translations already considered, thus obtaining a “spirally-oriented” list of copies of P .

At the end, for every element of our list, where the first is the one which is in the center of M , we recursively place the k points in every possible way.

step 3. Finally, the last part consists in checking the periodicity of the obtained configurations and the link between them and the periodicity induced by the tested polyomino.

What we have observed by using this program to look at the pseudo-square and the pseudo-hexagon of semi-perimeter ≤ 12 , is that the results are exactly that which we were expecting. In fact, the k -homogeneous configurations obtained are in agreement to the conditions of periodicity under described.

5 Conclusions

In this paper we have proved that the Decomposition Theorem (Theorem 2) holds for the diamonds class, and we have provided experimental confirmation that such a Theorem holds to pseudo-square and pseudo-hexagon polyominoes. We are then led to conjecture that such a Theorem can be extended to the whole class of exact polyominoes.

In the future we would also like to analyze the problem of reconstruction of homogenous configurations from their diamond-scan and then for every exact polyominoes-scan.

References

1. Beauquier, D., Nivat, M.: On Translating one polyomino to tile the plane. *Discrete Comput. Geom.* 6, 575–592 (1991)
2. Brlek, S., Frosini, A., Rinaldi, S., Vuillon, L.: Tilings by translation: enumeration by a rational language approach. *The Electronic Journal of Combinatorics* 13(1) (2006)
3. Brlek, S., Provençal, X.: An optimal algorithm for detecting pseudo-squares. In: Kuba, A., Nyúl, L.G., Palágyi, K. (eds.) *DGCI 2006. LNCS*, vol. 4245, pp. 403–412. Springer, Heidelberg (2006)
4. Nivat, M.: On a tomographic equivalence between $(0,1)$ -matrices. In: Karhumäki, J., Maurer, H., Păun, G., Rozenberg, G. (eds.) *Theory Is Forever. LNCS*, vol. 3113, pp. 216–234. Springer, Heidelberg (2004)
5. Frosini, A., Nivat, M.: Binary matrices under the microscope: a tomographical problem. *TCS* 370, 201–217 (2007)
6. Herman, G.T., Kuba, A.: *Discrete tomography: Foundations algorithms and applications*. Birkhauser, Boston (1999)
7. Nivat, M.: Sous-ensembles homogenes de \mathbb{Z}^2 et pavages du plan. *C. R. Acad. Sci. Paris. I* 335, 83–86 (2002)
8. Provençal, X.: *Combinatoire des mots, geometrie discrete et pavages*. Ph.D. thesis, D1715, Université du Québec a Montréal (2008)

Convex-Set Perimeter Estimation from Its Two Projections^{*}

Étienne Baudrier, Mohamed Tajine, and Alain Daurat

Laboratoire des Sciences de l'Image, de l'Informatique et de la Télédétection
CNRS UMR 7005, University of Strasbourg, Strasbourg, France
{baudrier,tajine}@unistra.fr

Abstract. This paper addresses the problem of extracting qualitative and quantitative information from few tomographic projections without object reconstruction. It focuses on the extraction of quantitative information, precisely the border perimeter estimation for a convex set from horizontal and vertical projections. In the case of a unique reconstruction, we give conditions and a method for constructing an inscribed polygon in a convex set only from the convex-set projections. An inequality on border perimeter is proved when a convex set is included in another one. The convergence of the polygon perimeter when the vertex number increases is established for such polygons. In the case of a multiple reconstruction, lower and upper bounds for the perimeter are exhibited.

Keywords: perimeter estimation, convex set, tomography, two projections, polygonal reconstruction.

1 Introduction

This paper addresses geometrical property estimation from convex-set projections without reconstructing the convex set. It is known that a convex set can be reconstructed from seven projections and from four projections with conditions on the angles [1]. So information of four projections is sufficient to geometrical property estimation. As the projection preserves the area, one projection is sufficient for the area estimation. On the contrary, the perimeter cannot be estimated from only one projection: Let P be a parallelogram with two sides parallel to Oy and with abscissa a for the first side and b for the second. It will have the same projection against Ox (a rectangle) and have an perimeter arbitrarily large, depending on the shearing of the parallelogram.

Thus we focus on the perimeter estimation from two projections in this paper. There are two cases whether the reconstruction is unique or not. If the reconstruction is unique, we propose a perimeter estimation from a polygon approximating the convex set with a small complexity. For this purpose, we first prove that the inclusion of convex sets implies the inferiority of their perimeters. Then, in a second section, the conditions on the projections that imply the convex-set inclusion are detailed. In a third part, the polygon reconstruction is studied. In a last part, the convex-set perimeter estimation based on a polygon is studied (construction and convergence) and bounds are computed where reconstruction is not unique.

^{*} This work was supported by the Agence Nationale de la Recherche through contract ANR-2010-BLAN-0205-01.

1.1 State of the Art

The perimeter estimation from two projections without reconstruction is not studied in the literature according to the authors. The characterization of unique reconstruction has been studied theoretically in [32] and with a statistical point of view in [4].

In all the paper, the sets, equalities and inequalities are defined modulo a set of measure zero (in the sens of the measure of Lebesgue) and all the functions are measurable.

The following notions are used in the sequel.

Definition 1 (Hypograph, epigraph). For a function with value in \mathbb{R} , its hypograph is the set $HG(f) = \{(x, y) \mid x \in \text{supp}(f) \text{ and } y \leq f(x)\}$ and its epigraph is the set $EG(f) = \{(x, y) \mid x \in \text{supp}(f) \text{ and } y \geq f(x)\}$.

Definition 2 (Convexity). Let C be a set in a real vector space. C is said to be convex if

$$\forall x, y \in C, \forall t \in [0, 1], (1 - t)x + ty \in C.$$

A function f is convex if $EG(f)$ is a convex set.

A function f is concave if $-f$ is convex, which is equivalent to $EG(f)$ is a concave set.

Remark 1. It is noticeable that the projections of a convex set along horizontal and vertical directions are concave functions, which is equivalent to: the hypographs of these projections are convex sets.

Definition 3 (Function comparison). Let $f : D_f \rightarrow \mathbb{R}^+$ and $g : D_g \rightarrow \mathbb{R}^+$ be two functions, we denote $f \preceq g$ if $D_f \subseteq D_g$ and $\forall x \in D_f, f(x) \leq g(x)$.

Our work exploits in sec. 3 the theorems of characterization and reconstruction proposed in [2]. Let introduce notations and recall the main characterization theorems quoted from [2].

Let $C \subseteq \mathbb{R}^2$ be a set such that $\lambda_2(C) < \infty$, where λ_2 is the two-dimensional Lebesgue’s measure. Let $\chi_C(x, y)$ be the characteristic function of C . Let λ_1 be the one-dimensional Lebesgue’s measure. From the Fubini’s theorem, the projections of $\chi_C(x, y)$ along horizontal direction:

$$f_X^C(y) = \int_{-\infty}^{\infty} \chi_C(x, y) dx = \lambda_1(\{x \mid (x, y) \in C\})$$

and vertical direction:

$$f_Y^C(x) = \int_{-\infty}^{\infty} \chi_C(x, y) dy = \lambda_1(\{y \mid (x, y) \in C\})$$

exist almost everywhere on \mathbb{R} and are integrable. Let $C_X = \{y \mid f_X^C(y) > 0\}$ and $C_Y = \{x \mid f_Y^C(x) > 0\}$ be the supports of f_X^C and f_Y^C respectively. As the sets $\{y \in C_Y \mid f_X^C(y) \geq x\}$ and $\{x \in C_X \mid f_Y^C(x) \geq y\}$ are measurable sets, the functions

$$f_{XY}^C(x) = \lambda_1(\{y \in C_Y \mid f_X^C(y) \geq x\})$$

and

$$f_{YX}^C(y) = \lambda_1(\{x \in C_X \mid f_Y^C(x) \geq y\})$$

exist for almost all x and y .

Remark 2. – $f_{X,Y}^C = f_Y^{HG(f_X^C)}$.
 – $f_{Y,X}^C = f_X^{HG(f_Y^C)}$.

Similarly, one can define

$$f_{YXY}^C(x) = \lambda_1(\{y \mid f_{YX}^C(y) \geq x\})$$

Definition 4. A pair of functions $g : I_1 \rightarrow \mathbb{R}^+, h : I_2 \rightarrow \mathbb{R}^+$ is called *unique*, (respectively *non-unique* (respectively *inconsistent*)) if there exists an unique set D (respectively different sets D (respectively no set D)) such that $D \subseteq I_1 \times I_2$, $f_X^D = g$ and $f_Y^D = h$

We then have the following theorem (detail and proofs can be found in [2]):

Theorem 1 (Characterization of unique, non unique and inconsistent projections). Let f_X^C and f_Y^C be two integrable positive functions such that

$$\int_{-\infty}^{\infty} f_X^C(y)dy = \int_{-\infty}^{\infty} f_Y^C(x)dx$$

1. f_X^C et f_Y^C are unique if and only if

$$\forall z > 0, \int_0^z f_{XY}^C(x)dx = \int_0^z f_{YXY}^C(x)dx$$

2. f_X^C and f_Y^C are non-unique if and only if

$$\forall z > 0, \int_0^z f_{XY}^C(x)dx \geq \int_0^z f_{YXY}^C(x)dx$$

and there exists a z for which strict inequality holds

3. f_X^C and f_Y^C are inconsistent if and only if

$$\exists z > 0, \int_0^z f_{XY}^C(x)dx < \int_0^z f_{YXY}^C(x)dx$$

Another unique characterization which will be useful is given in [3]:

Theorem 2 (Characterization with inverse). The set C is determined uniquely modulo null sets by its projections if and only if f_{XY}^C and f_{YXY}^C are inverses of each other (i.e. $(f_{XY}^C)^{-1} = f_{YXY}^C$).

Example 1. Let $a_X, a_Y \in \mathbb{R}^+$ and $a, b, c, d \in \mathbb{R}$ such that $a \leq b$ and $c \leq d$. Consider the functions $g_X = a_X \chi_{[a,b]}$ and $h_Y = a_Y \chi_{[c,d]}$ then

$$g_{XY} = (b - a) \chi_{[0, a_X]},$$

and

$$h_{YXY}^{-1} = a_Y \chi_{[0,d-c]}.$$

So, Theorem 2 implies that g_X, h_Y are unique if and only if $b - a = a_Y$ and $d - c = a_X$ and in this case, $g_X = f_X^R$ and $h_Y = f_Y^R$ where R is the rectangle $= [c, d] \times [a, b]$.

A theorem allowing the set C to be reconstructed is found in [2]

Theorem 3 (Convex-set reconstruction). *If C is uniquely determined by its projections f_X^C, f_Y^C then*

$$C = \{(x, y) \mid f_Y^C(x) \geq f_{XY}^C(f_X^C(y))\} \tag{1}$$

In all the following, a planar curve is considered as a function $\gamma : [0, 1] \mapsto \mathbb{R}^2$.

Definition 5 (Planar curve length). *Let $\gamma : [0, 1] \mapsto \mathbb{R}^2$ be a planar curve. The length of γ , $l(\gamma)$, is defined by*

$$l(\gamma) = \sup\left\{\sum_{i=0}^{n-1} d(\gamma(t_i), \gamma(t_{i-1})) \mid n \in \mathbb{N} \text{ and } 0 = t_0 < t_1 < t_2 < \dots < t_{n-1} < t_n = 1\right\}. \tag{2}$$

where d is the euclidean distance.

The perimeter of a convex set is then defined as follows:

Definition 6 (Perimeter of a convex set). *Let C a planar convex set, then there exists $\gamma_C : [0, 1] \rightarrow \mathbb{R}^2$ with $\partial(C) = \gamma_C([0, 1])$ (i.e. γ_C is a parametrization of the boundary $\partial(C)$ of C). The perimeter of C is noted $l(C)$ and is defined as $l(C) = l(\gamma_C)$.*

2 Convex Inclusion and Perimetric Inequality

In the following, straight lines are described by using two parameters $(r, \theta) \in \mathbb{P} = \mathbb{R}^+ \times [0, 2\pi)$.

So, let D be a straight line, D' be the straight line through the origin and perpendicular to D and $M = D \cap D'$. Then, the parameter r (respectively θ) is $d_2((0, 0), M)$, the distance of the origin to D (respectively the angle between D' and the X-axis). So, D will be denoted in the following by $D(r, \theta)$ (see Fig. 1 for illustration).

Theorem 4 (Crofton’s formula). *Let $\gamma : [0, 1] \mapsto \mathbb{R}^2$ be a planar curve. Then,*

$$l(\gamma) = \frac{1}{4} \int_{\mathbb{P}} n_{\gamma}(r, \theta) dr d\theta.$$

where $\mathbb{P} = \mathbb{R}^+ \times [0, 2\pi)$ and for all $(r, \theta) \in \mathbb{P}$, $n_{\gamma}(r, \theta) = \text{card}(\gamma([0, 1]) \cap D(r, \theta)) \in \mathbb{N} \cup \{\infty\}$ which is the number of intersection points of curve γ with the straight line $D(r, \theta)$.

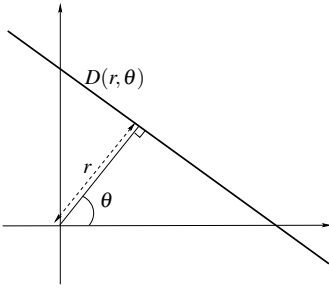


Fig. 1. Representation of a random straight line

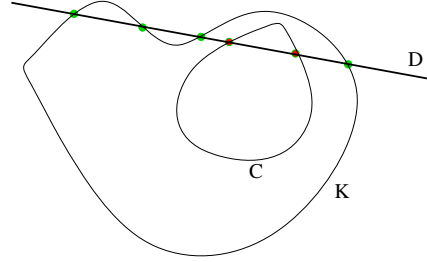


Fig. 2. Number of intersections of a random line D with the border of the sets C and K

Corollary 1 (Perimetric inequality). *Let K be a compact subset of \mathbb{R}^2 and C be a convex subset in \mathbb{R}^2 such that $C \subseteq K$. Then,*

$$l(C) \leq l(K).$$

Proof. For any straight line $D(r, \theta)$, as C is a convex set, $n_{\gamma_C}(r, \theta) = 0, 1, 2, \infty$. $n_{\gamma_C}(r, \theta) = \infty$ means that $C \cap D(r, \theta)$ is a segment of positive length. As C has a finite border perimeter, this situation occurs at most a countable number of times. Thus this situation has measure zero in \mathbb{P} .

In the other cases, we have $n_{\gamma_C}(r, \theta) \leq 2 \leq n_{\gamma_K}(r, \theta)$ (See Fig. 2 for illustration). So, the Crofton’s formula (Theorem 4) implies the result.

3 Projection Inclusion and Convex-Set Inclusion

The question addressed here is to know whether there is a link between the inferiority of horizontal and vertical projections on the one hand and the inclusion of convex sets themselves on the other. One implication is immediate:

Proposition 1 (inclusion, forward direction). *Let C and D be two measurable convex sets of \mathbb{R}^2 . Then,*

$$D \subseteq C \implies (f_X^D \preceq f_X^C \wedge f_Y^D \preceq f_Y^C).$$

The converse of Prop. 1 is not true in general. If we also assume that one has concave projections verifying $f_X^D \preceq f_X^C$ and $f_Y^D \preceq f_Y^C$, one cannot deduce that C and D are convex and $D \subseteq C$. Fig. 3 gives a counterexample. Although projections are concave and verify $f_X^D \preceq f_X^C$ and $f_Y^D \preceq f_Y^C$, the reconstructed set C is not convex and D is not included in C .

In the following, we prove that the converse is true under certain conditions i.e. those of Theo. 5.

To prove this theorem, we will first consider the case where the convex set D is a rectangle with one of its projections included in that of C . In a second step, we will prove the result in the case where C and D are both rectangles. In a third step, we use a decomposition of D in union of rectangles in order to prove the last result. Let first define an inscribed polygon.

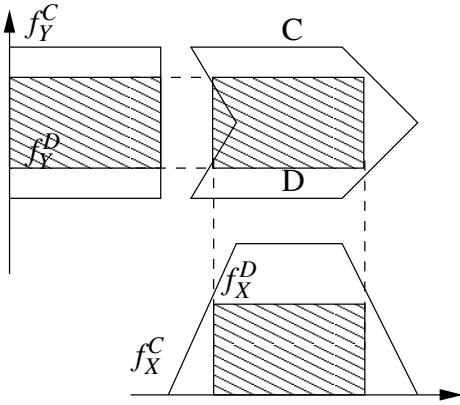


Fig. 3. Counterexample showing that inequalities on projections do not imply the convex-set inclusion

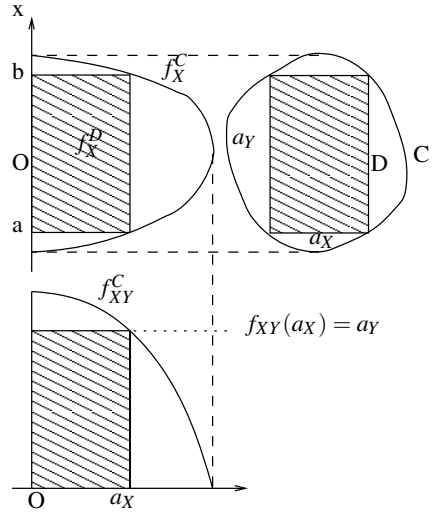


Fig. 4. Example of a rectangle inscribed in a convex set

Definition 7 (Inscribed polygon). A polygon P is said inscribed in a set F if its vertices belong to $\partial(F)$. A piecewise affine function f is said inscribed in a function g if $HG(f)$, which is a polygon, is inscribed in $HG(g)$.

Proposition 2 (Case of the rectangle with an included projection). Let C and D be two measurable convex sets of \mathbb{R}^2 and D be a rectangle whose sides are parallel to the directions of projection. Then,

$$f_X^D \preceq f_X^C \implies D \subseteq C.$$

The proof of Prop. 2 uses the following lemma:

Lemma 1. Let f_X^C, f_Y^C be unique projections of a convex set C and $g = a_X \chi_{[a,b]}$ be an inscribed function in the function f_X^C . Then there exists a rectangle D whose sides are parallel to the projection directions such that $f_X^D = g$ and f_Y^D is inscribed in f_Y^C . Moreover the rectangle D is inscribed in C .

Proof (Lem. 1). The conditions of Lem. 1 are equivalent to

- f_X^C, f_Y^C are unique
- $g \preceq f_X^C$
- $\exists a, b \in C_X, a < b$ and $g(a) = f_X^C(a) = g(b) = f_X^C(b) = a_X > 0$

As projections are unique, according to Theorem 2 we must have:

$$f_{XY}^C = (f_{YXY}^C)^{-1}.$$

By definition,

$$f_{XY}^C(a_X) = \lambda_1(\{y \mid f_X^C(y) \geq a_X\}),$$

so $f_{XY}^C(a_X) = a_Y$ and so $f_{YX}^C(a_Y) = a_X$. This means that

$$\lambda_1(\{y \mid f_{YX}^C(y) \geq a_Y\}) = a_X$$

and as f_{YX}^C is decreasing, it means

$$f_{YX}^C(a_Y) = a_Y$$

i.e.

$$\lambda_1(\{x \in C_X \mid f_Y^C(x) \geq a_Y\}) = a_Y \tag{3}$$

As f_Y^C is concave, $\exists c, d \in C_Y \mid c \leq d$ and $f_Y^C(c) = f_Y^C(d) = a_Y$. By (3), $d - c = a_X$. Therefore the rectangular function $h : x \mapsto a_Y \chi_{[c,d]}(x)$ is inscribed in f_Y^C . Thus, the rectangle $D = [a, b] \times [c, d]$ is inscribed in C and its projections are $f_X^D = g$ and $f_Y^D = h$. Moreover f_Y^D is inscribed in f_Y^C . \square

We can see that (under the hypothesis of the lemma 11) f_Y^D is inscribed into f_Y^C by construction. We introduce a definition reflecting this property:

Definition 8 (P-inscribed). Let f_X^C, f_Y^C and f_X^D, f_Y^D be projections respectively of a measurable convex set C and a rectangle D whose sides are parallel to the directions of projection. We say that D is p -inscribed in C if f_X^D (resp. f_Y^D) is inscribed in f_X^C (resp. f_Y^C).

We can now prove the proposition 2

Proof. As their projections are unique, C and D are characterized by Eq. (11). Let (x, y) be such that

$$f_X^D(y) \geq f_{XY}^D \circ f_Y^D(x),$$

show that this implies that

$$f_X^C(y) \geq f_{XY}^C \circ f_Y^C(x),$$

By hypothesis, one has

$$f_X^C(y) \geq f_X^D(y). \tag{4}$$

Show that $f_{XY}^D \circ f_Y^D(x) \geq f_{XY}^C \circ f_Y^C(x)$. Recall that f_Y^D is a rectangular function whose support is the interval $[a, b]$. In addition, projections are concave and, by definition, the projections f_{XY} are decreasing. So for the composition, functions $\psi^C = f_{XY}^C \circ f_Y^C, \psi^D = f_{XY}^D \circ f_Y^D(x)$ are concave. Distinguish cases according to the membership of x to $[a, b]$:

1. $x \in [a, b]$:

$$f_Y^C(a) = f_Y^C(b)$$

so

$$\psi^C(a) = \psi^C(b).$$

By concavity, one has therefore

$$\psi^C(x) \leq \psi^C(a) \tag{5}$$

On the other hand, as $f_Y^C(a) = f_Y^D(a)$ and $f_Y^C(b) = f_Y^D(b)$ and as these projections are concave on their support, one has

$$\psi^C(a) = \psi^D(a) \tag{6}$$

From (5) and (6), we deduce

$$\psi^C(x) \leq \psi^D(a) \tag{7}$$

now f_Y^D is constant on $[a, b]$ so

$$\psi^D(a) = \psi^D(x)$$

and replacing in (7) we obtain

$$\psi^C(x) \leq \psi^D(x) \tag{8}$$

2. $x \notin [a, b]$: $f_Y^D(x) = 0$ so $\psi^D(x) = 2$ so $\psi^C(x) \leq \psi^D(x)$

Thus

$$\forall x \in [-1, 1], f_{XY}^D \circ f_Y^D(x) \geq f_{XY}^C \circ f_Y^C(x) \tag{9}$$

By combining (4) and (9), we obtain that

$$f_X^C(y) \geq f_{XY}^C \circ f_Y^C(x),$$

so

$$(x, y) \in C$$

thus

$$D \subseteq C. \tag{□}$$

A consequence of Lem. 1 concerns the projection support length:

Corollary 2. *Let f_X^C, f_Y^C be unique projections of a convex set C , then $\lambda_1(\text{supp}(f_X^C)) = \sup(f_Y^C)$ and reciprocally.*

Proof. Let define a sequence of rectangular functions g_X^n inscribed in f_X^C with a height $\frac{1}{n}$ then

$$\lim_{n \rightarrow \infty} \lambda_1(\text{supp}(g_X^n)) = \lambda_1(\text{supp}(f_X^C)) \tag{10}$$

From Lem. 1, each g_X^n corresponds to a rectangle D_n such that $f_Y^{D_n}$ is inscribed in f_Y^C . Now $\lambda_1(\text{supp}(f_Y^{D_n})) = \frac{1}{n}$ tends to zero when $n \rightarrow \infty$, then $\max(f_Y^{D_n})$ tends to $\max(f_Y^C)$. With (10), we deduce that $\lambda_1(\text{supp}(f_X^C)) = \sup(f_Y^C)$. The result holds if f_X^C and f_Y^C are switched. □

We can now extend prop. 2 to the case where D is a subset of a rectangle inscribed in C . For this, introduce the following definition

Definition 9 (P-subinscribed). *D is called p -subinscribed in C if there is a rectangle D' such that $D \subseteq D'$ and D' is p -inscribed in C*

This definition implies (Prop. 1) that the projection of D is less than those of D' .

Proposition 3 (Inclusion by p -subinscribed). *Let $f_X^C, f_Y^C, f_X^D, f_Y^D$ be unique projections such as D is p -subinscribed in C then*

$$D \subseteq C$$

Proof (Prop. 3). From Def. 9 there is D' such that

$$f_X^D \preceq f_X^{D'} \preceq f_X^C, f_Y^D \preceq f_Y^{D'} \preceq f_Y^C$$

Thus, by prop. 2 we know that $D' \subset C$. On the other hand, as projections of D' are rectangular functions verifying

$$f_X^D \preceq f_X^{D'}, f_Y^D \preceq f_Y^{D'},$$

one has $D \subseteq D'$ so $D \subseteq C$ □

It is now possible to write a general theorem by decomposing D

Theorem 5 (Reciprocal inclusion). *Let $f_X^C, f_Y^C, f_X^D, f_Y^D$ be unique projections such as the set D is the convex hull of union of sets $(D_i)_{i \in I}$ p -subinscribed in C , then*

$$D \subseteq C.$$

Proof (thm 5).

$$\forall i \in I, \exists D_i | f_X^{D_i} \preceq f_X^{D_i}, f_Y^{D_i} \preceq f_Y^{D_i},$$

now the D_i are inscribed rectangles, so

$$\forall i \in I, D_i \subseteq D_i \text{ and } \forall i \in I, D_i \subseteq C$$

so

$$\bigcup_{i \in I} D_i \subseteq C$$

thus, if $CH(g)$ denotes the convex hull of g on its support,

$$D = CH\left(\bigcup_{i \in I} D_i\right) \subset CH(C) = C \quad \square$$

Corollary 3 (Inscribed polygon). *Let D_1, \dots, D_n be rectangles p -inscribed in a measurable convex set C and $f_X = CH(\max_i f_X^{D_i}, f_Y) = CH(\max_i f_Y^{D_i})$, these projections are unique and reconstruct a polygon inscribed in C*

4 Estimation of Convex-Set Perimeter

Let C be a convex set such that f_X^C, f_Y^C are unique projections and P be convex polygon which is inscribed in C . Considers $\mathfrak{V}(P) = \{p_0, \dots, p_{n-1}\}$ be the set of ordered vertices of P with $p_n = p_0$.

4.1 Unique Reconstruction

We first give an upper bound of the difference between the perimeters of C and P then we present a limit for the perimeter of P when the number of vertices tends to infinity. Let introduce the following notations: let $\gamma_K : [0, 1] \rightarrow \mathbb{R}^2$ a parametrization of ∂K where $K = C, P$. As P is inscribed in C we can define $0 = t_0 < t_1 < \dots < t_n = 1$ such that $\gamma_K(t_0) = p_0, \dots, \gamma_K(t_n) = p_n$ for $K = C, P$. We define

- $I_{axe} = \{i \mid \gamma([t_i, t_{i+1}]) \parallel Oaxe\}$ where $axe = X, Y$. These sets are defined to deal with the edges parallel to the axes Ox and Oy
- $I = \llbracket 1, n \rrbracket \setminus (I_X \cap I_Y)$

Let $i \in (I \cap I_X)$, we define

- $F_{X,i}^K = \{(y, f_X^K(y)) \mid y \in [y_{p_i}, y_{p_{i+1}}]\}$ where $K = C, P$
- the affine transformation $\mathcal{A}_{X,i}$ determined by $\mathcal{A}(F_{X,i}^P) = P_i$.
- the convex set $D_i = CH(P_i \cup C_i)$
- the convex set $M_{X,i} = CH(P_i \cup \mathcal{A}(F_{X,i}^C))$

In the same way, we also define for $i \in (I \cap I_Y)$, $F_{Y,i}^K, K_i, \mathcal{A}_{Y,i}, M_{Y,i}, D_i$.

Fig. 5 illustrates the notations introduced above.

Theorem 6. *We have the following inequality:*

$$l(C) \leq \sum_{i \in I} \min(l(\mathcal{A}(F_{X,i}^C)), l(\mathcal{A}(F_{Y,i}^C))) + \sum_{i \in I_X} l(F_{X,i}^C) + \sum_{i \in I_Y} l(F_{Y,i}^C)$$

Remark 3. This upper bound does not depend on the convex set C reconstruction

Lemma 2. $\forall i \in I \cup I_{axe}, D_i \subseteq M_{axe,i}$ with $axe = X, Y$

Proof (Lem. 2). To simplify the notation, we take $axe = Y$. Let $i \in I \cup I_Y$ and $x_0 \in \text{supp}(f_Y^P)$. As $P \subseteq C$ and C is convex, $(C \setminus P) \cap \{x = x_0\}$ is composed of at most two segments (which can be empty) that we note d_0^u and d_0^l : $(C \setminus P) \cap \{x = x_0\} = d_0^u \cup d_0^l$; u

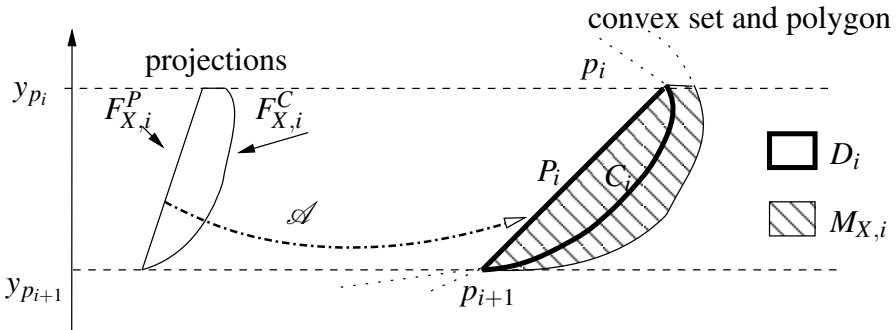


Fig. 5. The notations for Theo. 6 are presented on a part of the convex set C and the convex P

(resp. l) denotes the upper (resp. lower) segment position. Then the projection can be decomposed as follows:

$$f_Y^C(x_0) = f_Y^P(x_0) + l(d_0^u) + l(d_0^l)$$

For sake of simplicity, we suppose that the vertical position of the polygon edge P_i is on the upper side of the polygon. By construction,

$$(D_i \cap \{x = x_0\}) = \{(x_0, y) \mid y \in (d_0^u + \mathcal{A}(f_X^P(x_0)))\}$$

and

$$(M_{Y,i} \cap \{x = x_0\}) = \{(x_0, y) \mid y \in (d_0^u \cup d_0^l + \mathcal{A}(f_X^P(x_0)))\}$$

then

$$\forall x_0 \in \text{supp}(f_Y^P), D_i \cap \{x = x_0\} \subseteq M_{Y,i} \cap \{x = x_0\}$$

so

$$D_i \subseteq M_{Y,i} \quad \square$$

Let prove the theorem:

Proof (Theo. 6). Let $i \in I \cup I_Y$. Lem. 2 implies $D_i \subseteq M_{Y,i}$. Then by Cor. 11

$$l(D_i) \leq l(M_{Y,i})$$

i.e.

$$l(C_i) + l(P_i) \leq l(\mathcal{A}(F_{Y,i})) + l(P_i)$$

then

$$l(C_i) \leq l(\mathcal{A}(F_{Y,i}))$$

Thus if $i \in I$, this inequality holds with $axe = X$ and so

$$l(\gamma_C([t_i, t_{i+1}])) \leq \min(l(\mathcal{A}(F_{Y,i})), l(\mathcal{A}(F_{X,i})))$$

For $i \notin I$, the edge $\gamma_P([t_i, t_{i+1}])$ is parallel to an axis, e.g. Ox . Then the transformation \mathcal{A} comes down to a translation that conserves the length. Thus $l(\mathcal{A}(F_{Y,i})) = l(F_{Y,i}) \quad \square$

To demonstrate the polygon perimeter convergence toward the convex-set perimeter, we have to establish the following lemma.

Lemma 3. *Let f_X^C, f_Y^C be two unique projections of a convex set C and let f_X^P, f_Y^P be the unique projections of polygon P included in C . Then we can construct a polygon Q such that $P \subseteq Q \subseteq C$.*

Proof (Lem. 3). Let p_x, p_y be the coordinates of a breakpoint p of the piecewise linear function f_X^P . As P is included in C , $f_X^P \preceq f_X^C$. Let define f_X a rectangular function inscribed in f_X^C such that $f_X(p_x) = f_X^C(p_x)$. From Lem. 11 there exists an inscribed rectangle R_p in C such that $f_X^{R_p} = f_X$. As R_p is inscribed in C , it contains all the points of C whose abscissa is p_x and in particular, R_p contains the point p .

Let define $Q = CH(\cup_p R_p)$ where p are the breakpoints of P . Then Q is a convex polygon inscribed in C containing all the breakpoints of P . As Q is convex, it contains also P . □

Theorem 7 (Perimeter convergence). *Let f_X^C, f_Y^C be two unique projections of a convex set C then $\exists (P_n)_{n \in \mathbb{N}}$ a sequence of inscribed polygons in C such that $\lim_{n \rightarrow \infty} l(P_n) = l(C)$.*

The proof of Theo. 7 cannot be detailed in this paper due to the lack of space.

Thus, in the case of unique reconstruction, an estimation of the convex-set perimeter is made thanks to a polygonal approximation. The polygon construction exploits the reconstruction formula available for the unique projections. So, the method cannot be extended to the multiple reconstruction case. We propose for this case rough approximations.

A naive upper bound is given by the projection supports: $C \subseteq \text{supp}(f_Y^C) \times \text{supp}(f_X^C)$ then, by Cor. 1 $l(C) \leq 2(l(\text{supp}(f_Y^C)) + l(\text{supp}(f_X^C)))$. The upper bound proposed below is better than the naive bound only in the case of multiple reconstructions.

4.2 Multiple Reconstruction

Lower Bound

Definition 10 (Steiner’s symmetrized set). *Let C be a plane measurable convex set and f_Y its vertical projection. The Steiner’s symmetrized set $\text{sym}_Y(C)$ against Ox is defined by:*

$$\forall \alpha, \text{sym}_Y(C) \cap (x = \alpha) = \{(\alpha, y) \mid -\frac{1}{2}f_Y(\alpha) \leq y \leq \frac{1}{2}f_Y(\alpha)\}$$

Theorem 8 (Perimeter lower bound). *Let C be a measurable convex set \mathcal{C}_m^1 and $\text{sym}_Y(C)$ be its Steiner’s symmetrized set, thus one has:*

$$l(\text{sym}_Y(C)) \leq l(C)$$

Proof. One can parametrize C with two concave functions $x \mapsto f_1$ and $x \mapsto f_2$. As C belongs to the class \mathcal{C}_m^1 , f_1 and f_2 are concave and \mathcal{C}_m^1 , and thus, the perimeter of C can be defined as

$$\int_{-1}^1 \|C'\| = \int_0^1 \sqrt{1 + f_1'^2} + \sqrt{1 + f_2'^2}$$

For the symmetrized set, its two concave functions are equal and equal to $f_3 = (f_1 + f_2)/2 \in \mathcal{C}_m^1$. So its perimeter is

$$l(\text{sym}_Y(C)) = 2 \int_0^1 \sqrt{1 + f_3'^2} \tag{11}$$

$$= \int_0^1 \sqrt{4 + (f_1' + f_2')^2} \tag{12}$$

The following result is well-known:

$$\sqrt{4 + (a + b)^2} \leq \sqrt{1 + a^2} + \sqrt{1 + b^2},$$

which concludes the proof. □

Upper Bound. Assume that f_X^C, f_Y^C belong to the class \mathcal{C}_m^1 on their support D_x, D_y . As f_Y^C is null out of D_y and with positive value, there exists $x_0 \in D_y \mid f_Y^C(x_0) = 0$. With a parametrization of C with two concave functions f_1 et f_2 , one has $f_Y^C = f_1 - f_2$ and as $f_Y^C(x_0) = 0$, one has $f_1'(x_0) = f_2'(x_0)$. Moreover, as f_1 and f_2 are concave, their hypographs are under their tangents. It implies that C is between two straight lines

$$D_1 : y = f_1(x_0) + f_1'(x_0)(x - x_0)$$

and

$$D_2 : y = f_2(x_0) + f_2'(x_0)(x - x_0).$$

We note $d = f_Y^C(x_0)$, $c = f_1'(x_0) = f_2'(x_0)$, $a = l(D_y)$, $b = l(D_x)$ and we define a parallelogram

$$P = \{(x, y) \mid y \in D_x \text{ and } D_1(x) \leq y \leq D_2(x)\}.$$

Then $l(P) = 2 \left(d + \sqrt{a^2 + (ac)^2} \right)$, where c is the only unknown value. By construction of the parallelogram P , there is necessarily a point of C belonging to both of the vertical sides of P . So the slope c verifies $|c| \leq (b + d)/a$. The function

$$g : c \mapsto 2 \left(d + \sqrt{a^2 + (ac)^2} \right)$$

is even and growing on $[0, \frac{b+d}{a}]$, thus the maximum perimeter is

$$l(P) = 2 \left(d + \sqrt{a^2 + (b + d)^2} \right).$$

As $C \subseteq P$, $l(C) \leq l(P)$, so one has the following proposition:

Proposition 4 (Perimeter upper bound). *For a measurable convex set C whose frontier is \mathcal{C}_m^1 , the perimeter has an upper bound*

$$l(C) \leq 2(d + \sqrt{a^2 + (b + d)^2}) \tag{13}$$

where $d = \max(f_Y^C)$, $a = l(\text{supp}(f_Y^C))$, $b = l(\text{supp}(f_X^C))$.

One can easily see that the inequality comes down to the circumscribed rectangle perimeter when $d = l(D_x)$ (and the slope $c = 0$) thus this upper bound is interesting only when $d < l(D_x)$. Now a unique reconstruction implies that $d = l(D_x)$ so Prop. 4 is interesting only in the case of a multiple reconstruction.

5 Conclusion

In this paper, we present results regarding perimeter inequality and convex-set inclusion. In the case of unique reconstruction, we find conditions on the projections to construct an inscribed polygon. This has led to an algorithm that constructs polygon projection such as the polygon is included in the convex set. An upper bound is exhibited for the difference between the perimeter of the convex set and the polygon.

Moreover, the convergence toward the convex set perimeter has been proved. The case of multiple reconstruction is more difficult to tackle with because no reconstruction formula is available. Nevertheless we propose lower and upper bounds for the perimeter in this case. A perspective is to answer the question: if there is multiple reconstructions for a given pair of projections, do the reconstructions have the same perimeter ?

Acknowledgements. Alain Daurat, co-author of this article, died on June the 25th, 2010. This article is dedicated to his memory.

References

1. Gardner, R.: Geometric tomography. Encyclopedia of Math and its App., vol. 58. Cambridge University Press, Cambridge (1995)
2. Kuba, A., Volčič, A.: Characterisation of measurable plane sets which are reconstructable from their two projections. *Inverse Probl.* 4(2), 513 (1988)
3. Lorentz, G.G.: A problem of plane measure. *Am. J. Math.* 71(2), 417–426 (1949)
4. Milanfar, P., Karl, W.C., Willsky, A.S.: Reconstructing binary polygonal objects from projections: A statistical view. *CVGIP: Graph. Mod. and Im. Proc.* 56(5), 371–391 (1994)

Solving the Two Color Problem: An Heuristic Algorithm

Elena Barucci, Stefano Brocchi, and Andrea Frosini

Dipartimento di Sistemi e Informatica, Università di Firenze,
Viale Morgagni, 65 - 50134 Firenze - Italy
{barucci,brocchi}@dsi.unifi.it, andrea.frosini@unifi.it

Abstract. The 2-color problem in discrete tomography requires to construct a 2-colored matrix consistent with a given set of projections representing the number of elements of each color in each one of its rows and columns.

In this paper, we describe an heuristic algorithm to find a solution of the 2-color problem, that has been recently proved to be NP-complete. The algorithm starts by computing a solution where elements of different colors may overlap, and then it proceeds in searching for switches that leave unaltered the projections but remove the overlaps. Experimental results show that this heuristic approach finds a solution in a short computational time to almost all the randomly generated 2-color instances, and it provides for the remaining ones a high quality approximation.

Keywords: Discrete tomography, reconstruction algorithm, color problem.

1 Introduction and Definitions

Let us consider a matrix where each non-zero element can be chosen from a set of k different ones, that we choose to represent as if painted with k different colors. Such a matrix is called k -color matrix, and it is commonly used to model structures that can be thought as discrete sets whose minimal constituents are of k different types, such as crystals or computer images.

One of the most intriguing problems concerning colored matrices is the k -color that asks to perform in polynomial time a faithful reconstruction of a k -color matrix compatible with a given set of horizontal and vertical projections, i.e. from vectors containing the number of elements of each color, for each row and column.

The simplest case is the 1-color problem that can be solved in linear time (with respect to the dimension of the solution) by using a greedy strategy. Ryser proved that a necessary and sufficient condition for a solution to exist is, said H and V the horizontal and vertical projections, and defining \bar{H} as

$$\bar{h}_j = |\{h_i \in H : h_i \geq j\}|.$$

then the problem has a solution if and only if

$$\sum_{j=l}^n v_j \geq \sum_{j=l}^n \bar{h}_j, \quad \text{with } 1 \leq l \leq n.$$

Thanks to this property, the consistency of the one color problem can be verified even without attempting the reconstruction. Unfortunately in the formal theory of the problem there is no similar formula for the multiple color problems, not even relatively only to the sufficiency or to the necessity of a condition, apart few exceptions that assure consistency for a limited family of instances as in [6] or to applying Ryser's condition to all of the colors individually to eventually prove the absence of a solution (as described in the following).

In [11], the authors furnished a proof of the NP-hardness of the k -color problem, with $k \geq 6$, with two projections. Later and with different techniques, in [7], it was shown that the presence of three different types of atoms is sufficient to maintain the k -color problem NP-hard. Only recently, this result has been definitively extended to $k = 2$ in [9], by proving its equivalence with *Vertex Cover*.

However, the space of solutions of an instance of the reconstruction problem is, in general, really huge and quite impossible to control. Furthermore, two of these solutions may also share no points, as one can immediately realize if considers that all the permutation matrices of the same size have the same horizontal and vertical projections. This problem, called *uniqueness problem*, is of primary relevance, since practical applications usually require a faithful reconstruction of an unknown object. So, many different ideas has been carried on and, among them, one of the most promising is that of taking advantage of some a priori knowledge, to guide the reconstruction process towards a specific subclass of the solutions. Usually this knowledge is expressed in terms of geometric constraints that the final solution has to fulfill, such as connectiveness, convexity or adjacency of its elements (see for instance the uniqueness result in [10] for the subclass of convex binary matrices), but it can also be required, as an example, to be a typical member of a class of binary matrices having a certain Gibbs distribution [15].

These and similar problems that concern the retrieval of geometric information about an object, represented by an integer values matrix, fits in the area of the Discrete Tomography and are widely studied not only for their theoretical interests, but also for many practical applications: most of the mathematical techniques developed have applications in other fields such as image processing [18], statistical data security [14], biplane angiography [16], graph theory [1] and so on. As a survey of the state of the art of Discrete Tomography we can suggest the books [12] and [13].

Our studies focus on the problem 2-color, i.e. that of reconstructing, when possible, a 2-color matrix from its horizontal and vertical projections.

Even a so simple paradigm allows several applications that concern, as an example, all that cases where in a background environment there lies a main material pattern, and some impurities or void bubbles have to be detected; in particular, we mention techniques for colon's polyps detection, which needs the colon to be digitally straightened and then flattened in order to reveal polyps

that may be hidden from view behind the folds [2], algorithms for an efficient crystal detection such as DART [3] that has been applied to the detection of the shapes of nanocrystals consisting of few atoms of $ErSi_2$ that are artificially inserted on a narrow band of a semiconducting homogeneous matrix of SiC , or typical VLSI quality control, where quality means homogeneousness of the silicon layers of an integrated circuit on which a complex of electronic components and their interconnections are set.

From a theoretical point of view, 2-color has strong connections with problems concerning bipartite graphs as underlined in [6] [4] [8] [5] where reconstruction techniques based on a ‘divide et impera’ approach for subclasses of 2-color’s instances rely on results about flow charts or dynamic programming applied to graph theory. Unfortunately, these schemes restrict their usability only to some very specific subsets of instance of 2-color, and, sometimes, they act in pseudo polynomial time in one of the dimensions of the problem, being, at least, computationally unacceptable. The method we propose hereafter has an heuristic approach, but it has the advantage to be applicable to any instance of 2-color and to furnish an answer in short computational time.

More precisely, let M be a 2-color matrix of dimension $m \times n$ and whose elements are in $\Sigma = \{b, r, 0\}$, where b and r stand for the colors blue and red. We define the horizontal projections $H^b = (h_1^b, \dots, h_m^b)$ of M such that:

$$h_i^b = |\{M_{i,j} : M_{i,j} = b\}| \text{ with } j \leq n.$$

Analogously, we defined the vector of horizontal projections $H^r = (h_1^r, \dots, h_m^r)$, and the two vectors $V^b = (v_1^b, \dots, v_n^b)$, and $V^r = (v_1^r, \dots, v_n^r)$ of the vertical projections of A .

In the sequel we choose to visualize each 2-color matrix as a set of cells of two different colors on a squared surface. The element 0 of the matrix will correspond to the void cell.

Figure 1 shows a two colored matrix and its projections.

So, an instance of 2-color turns out to be a set of four vectors of horizontal and vertical projections, while a solution is a 2-color matrix compatible with all of them, if it exists.

The heuristic we present can be applied to a generic instance $I = (H^b, H^r, V^b, V^r)$ of the problem 2-color, and it starts by computing a 2-color matrix M that satisfies the projections of I , but having some positions where the two colors b and r may overlap. Such a matrix can be obtained by merging the solutions of the two related 1-color problems.

Then, for each positions where colors overlap, the algorithm scans the matrix M searching for an adequate configuration of elements that can be used to solve the conflict without changing the projections of M . Such configurations are commonly known in discrete tomography as *switching components*, while the action of changing the elements is called *switching operation*. Figure 2 shows two switching components involving a different number of elements, and the related switching operations that allow to move back and forth from one to the other. Note that each switching operation performed inside a matrix does not change its horizontal and vertical projections, as desired.

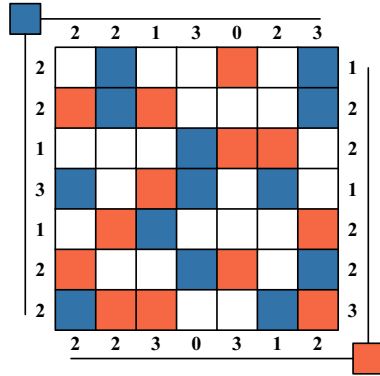


Fig. 1. A two colored matrix and its projections

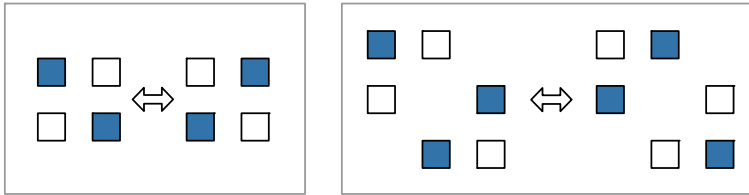


Fig. 2. Two switching components: (a) the simplest one involving four elements; (b) a more complex one that involves six elements

An exhaustive search for all the possible switching components would always furnish a solution to the problem, if possible, but it would require exponential time. In our algorithm, the search of the switching components is limited to some simple ones that can be determined efficiently. In the following we discuss how, assuming an uniform distribution of the colors in the matrix, it is very unlikely that such a component cannot be found; our experimentation proves that, even without considering this assumption, during the reconstruction process it is a rare case to find a matrix matching this configuration.

A simple property that is strongly used during the reconstruction process to keep the description of the algorithm simple, is the invariance of 2-color with respect to permutations of the rows and columns of the solution, and consequently of the related elements of the projections. So, we assume without loss of generality that rows and columns having specific properties lie in given positions.

2 The Heuristic Algorithm

Now, we first give an overview of the behavior of our algorithm, then we furnish the details.

As a first step, Ryser's necessary conditions, given in [17], are checked to guarantee that there is a solution for the 1-color instances $I^1 = (H^b, V^b)$ and $I^2 = (H^r, V^r)$ related to the elements b and r of M , separately. This check is also done for the one color problem obtained by summing the projections of the two colors; this is useful to detect some additional instances that cannot have a solution, for example because the sum of two horizontal projections of a row exceeds the matrix' width, even if the 1-color instances I^1 and I^2 would have a solution. If any of these instances is not satisfied, then we can state that also the instance I can not be satisfied as well. This is the only case where the algorithm is able to prove the non existence of a solution.

Otherwise, the algorithm computes a solution for the color blue (element b), say M^1 , and for color red (element r), say M^2 . The matrices are then merged in a matrix called M , that is not, in general, a 2-color matrix since in some positions the two colors may overlap. The matrix M can be easily obtained by starting from the void matrix and coloring each cell according to the presence, in the same position, of one of the two colors in M^1 or M^2 . If both colors appear in the same position, then we insert a new element in M that stands for the double color, say a *conflict*. This operation creates a matrix whose elements estimate those of a possible solution; to our knowledge, there is no other well known approach to compute a better estimation for this starting point.

Notice that, considering each conflict both as a blue and a red element, then M satisfies the instance I . Our goal is to find switches in M that remove the conflicts leaving unchanged the projections.

The algorithm at this point iterates through the conflicts, and for each one it searches for switching components of three different types described below. We show in the following how there is an efficient method to determine if a switch of these three types can be applied. If the procedure finds an applicable switch of type 1 or 2, then this is used to create a matrix with the same projections but with less conflicts. If the algorithm determines that switches of type 1 and 2 cannot be applied, but a switch of type 3 exists, then it applies the latter to obtain a matrix with equal projections and the same number of conflicts, but with a conflict in a different position. In this case, the procedure selects a submatrix of M and searches recursively switches of type 1 – 3 in this reduced structure.

If all of the switching operations are not sufficient to obtain a solution with no overlaps, the algorithm start over by computing a different solution for the 1-color problems and hence generating another initial matrix M . This is often sufficient to find a solution if the first iteration fails. The number of maximum iterations of this step can be tuned depending on the available time. In our tests, we have verified that increasing the number of iterations to more than 4 or 5 increases only very lightly the percentage of solved instances, at the cost of a slower computational process. Hence, we considered the limit of 5 iterations to be a suitable choice for the experimentation.

Algorithm. RECONSTRUCTION

- 1 Check if the 1-color instances $I^1 = (H^b, V^b)$, $I^2 = (H^r, V^r)$ and $I^3 = (H^b + H^r, V^b + V^r)$ are consistent; this can be done by applying Ryser's condition and assuring that every horizontal and vertical projection is not greater than the width or the height, respectively, of the output matrix. If the projections of any of these instances are inconsistent, then give FAILURE and halt.
- 2 Solve 1-color on the instances I^1 and I^2 , using a standard greedy strategy, as described in [17], obtaining the matrices M^1 and M^2 , respectively. Then merge the two solutions into the matrix M : the merging process initializes all the elements of M to 0, and then changes its value according to the presence of a color in the same position of M^1 or M^2 . If some colors overlap in a position, then this position is marked with an extra symbol that indicates a conflict of colors.
- 3 For each element c having a conflict
 - 3a Attempt to execute a switching operation of type 1 involving one color of c , as explained in Paragraph 2.1; if it exists, goto step 3;
 - 3b Attempt to execute a switching operation of type 2 involving one color of c , as explained in Paragraph 2.2; if it exists, goto step 3;
 - 3c Attempt to execute a switching operation S of type 3 involving one color of c in order to move the conflict in another position, as explained in Paragraph 2.3. As described in the following, this operation may involve the recursive application of steps 3a and 3b on a particular submatrix of M .
 - 3d If no switches of type 1, 2 or 3 can be executed, then exit loop in step 3.
- 4 If a matrix without conflicts has not been computed, then
 - 4a If this step has been executed less than p times, goto step 2 and generate different solutions M^1 and M^2 for the 1-color instances I^1 and I^2 ; these can be created by applying Ryser's algorithm to a random permutation of the arrays H^k and V^k , and by reorganizing the columns and rows of the output matrix M^k accordingly.
- 5 If an exact solution was found, return it, otherwise return, of the computed matrices, the one containing less conflicts.

2.1 Type 1 Switching Components

Assume without loss of generality that the element in $(1, 1)$ is a conflict. Assume that, for some k_1 , there are $k_1 - 1$ columns with a 0 in row 1, and without loss of generality consider that the indexes of these columns are in the interval $2 \dots k_1$. Notice that, since the projections are consistent, $k_1 > 2$. Symmetrically, consider the rows $2 \dots h_1$ to be the only ones with the first element equal to 0.

If any element (i, j) in the rectangle $(2, 2) \dots (h_1, k_1)$ is different from 0, then the procedure can solve the conflict by applying the switching operator on $\{(1, 1), (1, j), (i, 1), (i, j)\}$ and on color $m_{i,j}$.

We can observe that, if k_1, h_1 are proportional to m, n , and assuming uniformity in M , the probability to have a switch of this kind is proportional to

$P(0)^{mn}$. The experimental results show that in most of the cases this switching operation is sufficient to solve the conflict.

2.2 Type 2 Switching Components

If there are no switching components of type 1, then the algorithm proceeds by searching switching components of type 2, and that involve involve six or eight elements at a time. A further preprocessing stage on M is now needed, where rows and columns are organized in different groups, in order to easily define the procedure.

Let us consider the columns of index greater than k_1 such that at least one element in row $i > h_1$ is in position before a given index k_6 . Symmetrically, say that every row containing at least a zero in index $h > k_1$ is in position less than an index h_6 .

Now create a partition on all columns of index k , with $k_1 < k < k_6$, in four different groups. Each column is assigned to a different group depending on the colors found in positions $1...h_1$ as described in the following points:

- Group 1: all the columns that contain only the value 0 in positions $2...h_1$
- Group 2: all the columns that contain in positions $2...h_1$ at least an element b but no r , and an element b in position 1
- Group 3: all the columns that contain in positions $2...h_1$ at least an element r , but no bs , and an element r in position 1
- Group 4: all the other columns, i.e. all of those containing both an element b and an element r in positions $1...h_1$

Assume without loss of generality that the columns of group g are in indexes j such that $k_{g+1} \leq j < k_{g+2}$, for some apposite k_2, k_3, k_4, k_5 . Execute the same grouping on rows, considering the elements in columns $1...k_1$, and define consequently the indexes h_2, h_3, h_4, h_5 . Such representation is depicted in an example in figure 3.

Define as $X(a, b)$ the submatrix of M that is obtained by intersecting columns of indexes $k_a...k_{a+1} - 1$ with the rows of indexes $h_b...h_{b+1} - 1$. The algorithm is able to determine a switch of type 2 if any of the following zones contains at least one 0:

$$X(3, 3), X(4, 4), X(5, 3), X(5, 4), X(5, 5), X(3, 5), X(4, 5)$$

We describe the case where there exists an element 0 in position (i, j) of zone $X(5, 5)$; the other cases are symmetric or trivial. So, we search for two elements (i_2, j) and (i, j_2) of the same color, w.l.g. say b , such that $i_2 \leq h_1$ and $j_2 \leq k_1$. We distinguish two cases:

1. $i_2 > 1$ and $j_2 > 1$. In this case, a switching component for the element b that solve the conflict is $\{(1, 1), (1, j_2), (i, j_2), (i, j), (i_2, j), (i_2, 1)\}$.
2. $i_2 = 1$ and $j_2 > 1$ (a symmetrical case is $j_2 = 1$). Here, a switching component for b involves the elements $\{(1, j_2), (i, j_2), (i, j), (1, j)\}$.

Then we can surely find an element in position (i_3, j) having value r or a conflict, and execute the switching operation on color r in the elements $\{(1, 1), (i_3, 1), (i_3, j), (1, j)\}$

Note that thanks to the definition of X_5 , there is always at least an element that satisfies the cases 1 or 2.

Assuming uniformity, the probability to be unable to find a switch are proportional to $(P(0))^{mn}$ if $k_6 - k_2$ and $h_6 - h_2$ are proportional to m and n . While in the previous switch type this chance quickly goes to 0 if the matrix contains many zero elements, in this case it tends to 0 if the red and blue elements are dense.

Hence we guarantee a high probability of solving the problem in each case where the elements of the matrix are uniformly distributed.

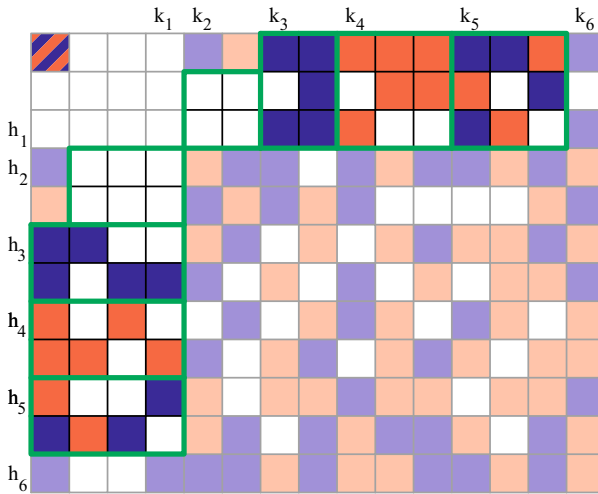


Fig. 3. Indexes k and h

2.3 Type 3 Switching Components

If a switch of type 2 is not found, we consider the grouping of rows and columns described in the previous section, and we search for a third type of switching components that does not solve the conflict but that moves it in a different position; then the algorithm attempts to remove the new conflict with type 1 and type 2 switching operations, using only a submatrix of M where these two types of switching components may lie. Since at least one row and one column are removed, this is sufficient to avoid infinite loops.

The procedure chooses, if available, one element from any of the following:

- An element r in the matrix $X(3, 3)$
- An element b in the matrix $X(4, 4)$
- Any element in one of the matrices $X(5, 3), X(5, 4), X(5, 5), X(3, 5), X(4, 5)$

We describe the case where an element in $X(5, 5)$ is selected, as all other result can be inferred by symmetry.

Let the selected element be in position (i, j) , with $k_5 \leq j < k_6$ and $h_5 \leq i < h_6$. Search for two cells (i_2, j) and (i, j_2) both of color b such that $0 < i_2 \leq h_1$ and $0 < j_2 \leq k_1$. At this point execute a switching operation for color b involving the elements $\{(1, 1), (1, j_2), (i, j_2), (i, j), (i_2, j), (i_2, 1)\}$. It is easy to see that this operation moves the conflict from position $(1, 1)$ to position (i, j) .

Now, we consider the submatrix formed by the intersection of columns $k_2, \dots, k_6 - 1$ and of rows $j_2, \dots, j_6 - 1$, and attempt to resolve the conflict of the element (i, j) throughout the three types of switching operations in this submatrix.

We have already pointed out that a matrix must have a very particular structure to avoid switching components of types 1 and 2; furthermore, if a switch of type 3 is applied, in order to be unable to solve the conflict, also the computed submatrix must have the same structure, and so the probability not to find a solution strongly decreases. The switching research operation also fails if no elements for the switching operation is found, i.e. there is no r element in matrix $X(3, 3)$, no b element in matrix $X(4, 4)$ and the matrices $X(5, 3)$, $X(5, 4)$, $X(5, 5)$, $X(3, 5)$, $X(4, 5)$ are all empty. Notice that matrices $X(3, 3)$ and $X(4, 4)$ cannot either contain the element 0, otherwise a switch of type 2 would have been applied. Also this case does not seem common as, again, this is a very peculiar configuration.

3 Generating Instances of 2-Color

To verify the performances of *Reconstruction*, we tested it on a large number of randomly generated instances of 2-color. Up to our knowledge there are no algorithms to generate them uniformly, hence we chose four different methods, each one related to a different aspect we desired to analyze. Our algorithm behaves extremely well for all the generated instances, even if some remarkable differences can be found for different generation methods.

Since we have no reason to give priority to problems with specific properties, the instances are generated with uniform probability on all the possible problems. The method *Gen1* aims to obtain an uniform probability distribution for the input projections, while the methods *Gen3* and *Gen4* instead create instances that have an uniform distribution of colors in an eventual solution matrix. The method *Gen2* is similar to *Gen1*, but in this case the uniformity assumptions are not considered in order to generate more problems with a solution, as described hereafter.

The generation procedures are simple to implement, and hence the experiments can be easily repeated and the results verified. By applying the algorithm to a very high number of problems, it is very unlikely that another run of the experimentation could furnish some significantly different results.

The first method, say *Gen1*, generates instances that are at most uniform in the domain of the integer vectors: at first we randomly generate H^b and H^r [resp. V^b and V^r] such that for each $i \leq n$ [resp. $j \leq m$] it holds $h_i^b + h_i^r \leq n$ [resp. $v_j^b + v_j^r \leq m$].

If after this operation the sums of the elements in the two arrays do not match, then a series of balancing operations is done. Assume without loss of generality that V^b has an element sum greater than H^r . In this case the algorithm select randomly an element $V_i^b > 0$ or an element $H_i^b < n$. In the first case, the element is decreased by one, otherwise it is increased of one. The procedure is iterated until the array sums become consistent.

A problem with *Gen1* is that it creates many instances that are provably unsolvable thanks to Ryser's condition, hence the solving algorithm can be applied only to a very limited portion of the generated problems. In *Gen2* the generation procedure avoids this problem by initializing the arrays with values that are the *average* of two integers randomly chosen with uniform distribution on the domain.

With this approach the distribution of the generated projections has a smaller variance, and, as discussed in related works as [6], this enhances the possibility of the existence of a solution.

The third method *Gen3* consists in generating a 2-colored matrix where each element is colored randomly; each color is chosen with equal probability, as the experimentation also proved that there are no significant changes in the results by altering the color ratio of red and blue. The projections extracted from this matrix represent the arrays on which the reconstruction will be attempted. This generation method has the important property to guarantee that each input set of arrays actually corresponds to a colored matrix; hence, if the algorithm fails to solve the problem, this cannot be caused by the lack of a solution.

Finally, the method *Gen4* extracts the projections from a 2-colored matrix obtained through a merging operation of two randomly generated 1-colored matrices relative to colors blue and red. The approach is similar to the one in *Gen3*, but since the matrix from where the projections are taken may contain some conflicts, then also problems with no solution can be generated.

4 Results

We have tested our algorithm on instances generated with the four described methods. The size of the instances has been varied in order to better understand its relation with the number of instances that can be solved, and to have some hints on how *Reconstruction* asymptotically behaves for large size matrices. The number of tests vary from 10^6 for 5×5 and 10×10 matrices, to 5000 for 100×100 problems. The instances are grouped in three categories: those where the algorithm is not able to find an exact solution, say *unsolved instances*, those where the algorithm proves the non satisfiability thanks to Ryser's conditions, say *provably unsolvable instances*, and those for which an exact solution has been computed, say *solved ones*.

In table [1A](#) we can see the percentage of problems that the algorithm was not able to solve, but for which it furnished only an approximation. The probability of this case goes to 0 very quickly with the size of the matrix, and for generation methods 3 and 4 even in the 10×10 problems an unsolvable instance was not

Table 1. Result table**A. Unsolved problems**

G. method	5×5	10×10	20×20	50×50	100×100
1	0.60%	0.69%	0.64%	0.59%	0.43%
2	0.24%	0.01%	0.005%	0	0
4	0.04%	0	0	0	0
3	0.001%	0	0	0	0

B. Average relative errors

G. method	5×5	10×10	20×20	50×50	100×100
1	0.23%	0.17%	0.10%	0.09%	0.08%
2	0.03%	0.002%	3×10^{-6}	0	0
4	0.007%	0	0	0	0
3	0.001%	0	0	0	0

C. Provably unsolvable problems

G. method	5×5	10×10	20×20	50×50	100×100
1	77.99%	91.00%	95.42%	97.82%	98.69%
2	19.98%	10.19%	4.27%	2.07%	1.52%
4	50.27%	44.97%	24.72%	0.90%	0
3	0	0	0	0	0

found in one million problems. *Gen1* seems to create the instances that are harder to solve. For a large subset of this family of instances, the algorithm proves that no solution exists thanks to Ryser's conditions. Probably, also good part of the instances for which the algorithm is unable to give an answer have to solution at all.

The average error of the solutions furnished by the algorithm is shown in Table [1](#) B. These ratios are obtaining by dividing the total number of overlapping cells by the global number of cells placed in all output solutions. In this computation only solved and unsolved problems are considered, as for the provably unsolvable problems no reconstruction is attempted. The average error seems to tend to zero with the growth of the problem size, even in *Gen1*, where the number of unsolved problems is very high.

Since the proposed algorithm obtains a very low number of unsolved instances, it is possible to use it to make statistics on how many problems with given properties do not have an existing solution. In Table [1](#) C we can see the percentage of problems that are provably unsolvable. With *Gen1*, this tends to be very high for big matrices; on the other hand, for *Gen2* and *Gen4*, this quantity quickly tends to 0. For instances generated with *Gen3* this number is clearly 0, as in this case a solution always exists.

A few words should be spent about time complexity. Even if the worst case complexity of the algorithm is higher, the experimental average complexity results to be approximately $O(mn)^2$; by tuning the parameters of the algorithm and settling for a weaker approximation, we could bound the worst case complexity of the procedure to this function, or we could greatly improve the average execution time.

Anyway, at the moment the processing time seems to be more than acceptable: even for a 100×100 matrix, and using an old Athlon 2600 processor, the problem can be solved in times ranging from 0.5 to 1 seconds.

5 Conclusions

Even if 2-color is an NP-hard problem, the algorithm we furnish gives excellent results: for several generation methods the number of solved instances is very close to the totality. We could suppose that the set of NP-hard instances are only a small subset of all the instances and have very peculiar properties. The results described in this paper suggest that the search of theoretical results about properties of the problem, regarding both necessary or sufficient conditions for a solution to exist, could be an interesting development. We suppose that by such conditions, most of the problems could be easily identified either as instances that admit a solution or as instances having no solution.

Many further research lines arise from this work. As a first idea, we could work on what is the 'weak point' of the algorithm: there is no non-trivial method to prove that a given problem has no solution. In fact, we suppose that many of the unsolved problems actually do not have a solution, as the greatest number of these instances appear in a generation method that creates many provably unsolvable problems. By finding some heuristic that allows us to prove for many cases that a solution doesn't exist, we could decrease the number of unsolved instances, and reduce to a minimum the NP-hard core of the problem.

Another possible development could be to extend the used techniques to problems with more than two colors. In this case, probably several major modifications of the algorithm will be necessary.

In order to allow an easy comprehension of the procedure and an easy testing of the algorithm, the procedure has been implemented in Javascript and can be found online at

www.researchandtechnology.net/discretetomography/3colorproblem/3colorsolver.html

References

1. Anstee, R.P.: Invariant sets of arcs in network flow problems. *Discrete Applied Mathematics* 13, 1–7 (1986)
2. 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: *Proceedings of SPIE Medical Imaging 2002: Image Processing*, San Diego, USA, vol. 4681, pp. 713–721 (2002)

3. 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)
4. 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)
5. Brocchi, S., Frosini, A., Picouleau, C.: Reconstruction of binary matrices under fixed size neighborhood constraints. *Theoretical Computer Science* 406(1-2), 43–54 (2008)
6. Brocchi, S., Frosini, A., Rinaldi, S.: Solving some instances of the two color problem. In: Brlek, S., Reutenauer, C., Provençal, X. (eds.) *DGCI 2009. LNCS*, vol. 5810, pp. 505–516. Springer, Heidelberg (2009)
7. Chrobak, M., Dürr, M.: Reconstructing polyatomic structures from X-rays: NP completeness proof for three atoms. *Theoretical Computer Science* 259, 81–98 (2001)
8. Costa, M.-C., de Werra, D., Picouleau, C., Schindl, D.: A solvable case of image reconstruction in discrete tomography. *Discrete Applied Mathematics* 148(3), 240–245 (2005)
9. Dürr, C., Guíñez, F., Matamala, M.: Reconstructing 3-colored grids from horizontal and vertical projections is NP-hard. In: Fiat, A., Sanders, P. (eds.) *ESA 2009. LNCS*, vol. 5757, pp. 776–787. Springer, Heidelberg (2009)
10. Gardner, R.J., Gritzmann, P.: Discrete tomography: determination of finite sets by X-rays. *Trans. Amer. Math. Soc.* 349, 2271–2295 (1997)
11. Gardner, R.J., Gritzmann, P., Pranenberg, D.: On the computational complexity of determining polyatomic structures by X-rays. *Theoretical Computer Science* 233, 91–106 (2000)
12. Herman, G.T., Kuba, A.: *Discrete tomography: Foundations algorithms and applications*. Birkhauser, Boston (1999)
13. Herman, G.T., Kuba, A.: *Advances in Discrete Tomography and Its Applications*. Birkhauser, Boston (2007)
14. Irving, R.W., Jerrum, M.R.: Three-dimensional statistical data security problems. *SIAM Journal of Computing* 23, 170–184 (1994)
15. 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)
16. Prause, G.P.M., Onnasch, D.G.W.: Binary reconstruction of the heart chambers from biplane angiographic image sequence. *IEEE Transactions Medical Imaging* 15, 532–559 (1996)
17. Ryser, H.J.: Combinatorial properties of matrices of zeros and ones. *Canadian Journal of Mathematics* 9, 371–377 (1957)
18. 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)

Approximating Bicolored Images from Discrete Projections

Fethi Jarray* and Ghassen Tlig

¹ Laboratoire CEDRIC, 292 rue Saint-Martin, 75003 Paris, France

² Al-Imam University, Riyadh, Kingdom of Saudi Arabia

fethi.jarray@cnam.fr, tlik@yahoo.fr

Abstract. We study the problem of reconstructing bicolored images from their discrete projections that is the number of pixels of each color lying on each row and column. The problem is well known to be NP-complete so, we study a restricted case (with bounded projections) and present an approximating algorithm based on a max-flow technique for the general case.

Keywords: Discrete Tomography, Image Reconstruction, Heuristics.

1 Introduction

Discrete Tomography (DT) deals with the reconstruction of digital image from its horizontal and vertical line sums. Digital images are most commonly represented by integer matrices. Let A be a binary matrix of size $m \times n$, we denote by $h_i = \sum_{j=1}^n A_{ij}$ the number of ones on row i and by $v_j = \sum_{i=1}^m A_{ij}$ the number of ones on column j . The vectors $H = (h_1, \dots, h_m)$ and $V = (v_1, \dots, v_n)$ are respectively called the horizontal and the vertical projections. The problem of reconstructing a binary matrix from orthogonal projections, denoted $MB(H, V)$, is defined as follows: Given two vectors H and V , we search to reconstruct a binary matrix consistent with these projections. It is well known that this problem is polynomial [25].

A k -colored image is an image where each pixel (or cell) is either uncolored or colored by one from a given set of k colors [8,15]. The projections of such an image consist of the number of pixels of each color on each line. The problem of reconstructing a k -colored image is defined as follows: Given the orthogonal projections of each color, we search to reconstruct a k -colored image consistent with these projections. For $k = 1$, the problem is equivalent to the binary matrix reconstructing problem. In general, the reconstruction of k -colored images is equivalent to k binary matrix reconstructing subproblems coupled by the exclusiveness constraint: A cell takes value 1 at the most in one subproblem.

The problem of reconstructing k -colored images arises on a number of applications, including industrial nondestructive testing [6], medical imaging [18,26,24]

* Corresponding author.

and workforce scheduling [21,20]. Gardner et al. [14] proved that the reconstruction of k -colored images is NP-complete for $k > 3$. Chrobak and Dürr [8] proved that the problem is NP-complete even for $k = 3$. Recently, Dürr et al. [12] proved that the reconstruction of two colors image is also NP-complete. Jarray [22] provided a lagrangean approach to reconstruct bicolored images from discrete orthogonal projections.

Several problems are at least as difficult as the reconstruction of bicolored images [5,9]. We will mainly focus in this paper on approximating bicolored images. The results can be easily extended to the k -colored images.

The remainder of this paper is organized as follows. In Section 2, we introduce some definitions and notations. In Section 3, we show that the binary images reconstruction problem can be solved using a network flow model, by solving a max flow problem. In Section 4, a special case of bicolored images is described. In this case the projection data are bounded and a polynomial time algorithm is given. In Section 5, we provide a heuristic algorithm for the general case based on a min-cost max-flow model. Finally, numerical results are presented and discussed in the last section.

2 Definitions and Notations

We suppose that for a bicolored image, the set of pixel values consists of only three elements: 0 (or "colorless"), 1 (or "color a") and 2 (or "color b"). The related consistency problem to bicolored image is defined as follows:

Instance: Integral vectors: $H^a \in N^m, V^a \in N^n, H^b \in N^m$ and $V^b \in N^n$.

Question: Is there a bicolored image respecting the projections (H^a, V^a) for color a and (H^b, V^b) for color b ?

Definition 1. Let x and y be two $m \times n$ matrices. We define the conflict $conf(x, y)$ between x and y as $conf(x, y) = \sum_{i=1}^m \sum_{j=1}^n x_{ij}y_{ij}$.

If x and y are binary, $conf(x, y)$ is the number of cells with value 1 on both matrices (overlapping 1's).

We introduce the operator \oplus between two binary matrices to build a bicolored image where color 'a' is associated with the 1's of one matrix and color 'b' with the 1's of the other.

Definition 2. Let x and y be two binary matrices, $S = x \oplus y$ is a bicolored image such that the cell (i, j) has the color 'a' (resp. 'b') if $x(i, j) = 1$ (resp. $y(i, j) = 1$).

Two bicolored images are equivalent if they have the same projections. Similarly to the switching operations in binary matrix [25], we define the following interchange operations to pass from an image to an equivalent image having

less number of conflicts: $\begin{bmatrix} & ab \\ ab & \end{bmatrix} \Rightarrow \begin{bmatrix} a & b \\ b & a \end{bmatrix}, \begin{bmatrix} & a \\ ab & \end{bmatrix} \Rightarrow \begin{bmatrix} a & \\ b & a \end{bmatrix}$. Clearly, carrying out an interchange operation does not modify the row and column sums and leads to

an equivalent image. Unlike the binary matrix reconstruction problem, it is not obvious that if two images are equivalent then one image can be transformed into the other by a finite sequence of interchange operations. The interchange operations will be used to solve the polynomial particular case.

We call a colored cell as a cell colored by 'a' or 'b' and a conflict cell as a cell colored by both 'a' and 'b'.

3 Flow Models to $MB(H, V)$

We recall two flow models for $MB(H, V)$ and in particular for both $MB(H^a, V^a)$ and $MB(H^b, V^b)$ to use later in the heuristics.

3.1 Max-flow Associated Problem

The fact that the binary reconstruction problem can be solved as a max flow problem was already demonstrated by Gale [13] and later used by Anstee [1], Slump et al. [28] and Salzberg et al. [27].

We will assign a max-flow problem to the binary matrix reconstruction problem $MB(H, V)$. We reconstruct a bipartite graph $G(R, C, E)$ where $R = \{r_i, i = 1, \dots, m\}$ represents the rows and $C = \{c_j, j = 1, \dots, n\}$ represents the columns (see Figure 2). We add to $G(R, C, E)$ two nodes: a source s and a sink t . There is an arc from s to every row node r_i with capacity h_i which is the horizontal projection of row i . By symmetry, there is an arc from every column node c_j to t with capacity v_j which is the vertical projection of column j . There is an arc from every pair of row node r_i and column node c_j . These arcs have a unit capacity and correspond to the cells of the matrix to reconstruct. Thus the problem $MB(H, V)$ is equivalent to the max-flow problem in G . $MB(H, V)$ admits a solution if and only if the maximum flow from the source to the sink is of value $\sum_{i=1}^m h_i = \sum_{j=1}^n v_j$. Since the capacities are integers, there exists an optimal integer flow. A solution to $MB(H, V)$ is computed by affecting to each cell (i, j) the flow on the corresponding arc (r_i, c_j) .

3.2 Min-cost Max-flow Associated Problem

As mentioned by Wang and Zhang [29] and Barcucci et al. [2], the problem of reconstructing binary matrices has an exponential number of admissible solutions. The best solution is chosen by adding an objective function to the reconstructing problem. In several practical applications, we have some a priori information about the matrix to reconstruct. For example, we search a solution to be as different as possible from a given binary matrix w . w is called the cost matrix. In such case, the objective is to minimize the conflict between w and the solution to $MB(H, V)$ (see Definition 1). The reconstruction problem is equivalent to a min-cost max-flow problem in the associated graph $G(R, C, E)$ where the arc (r_i, c_j) is with cost w_{ij} .

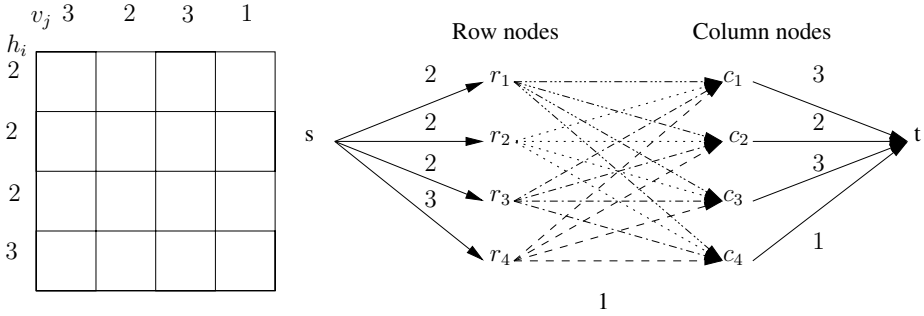


Fig. 1. $MB(H, V)$ and the associated max-flow in $G(R, C, E)$

Suppose that we have a solution y to the problem $MB(H^b, V^b)$, i.e. respecting the orthogonal projections of color b . If we set $w_{ij} = y_{ij}$, the min-cost max-flow problem associated to $MB(H^a, V^a)$ gives a solution to $MB(H^a, V^a)$ minimizing the conflict with y .

The use of a minimum cost flow algorithm for finding a solution which minimizes the number of conflicting ones with a second image was described by Batenburg [34]. For a bicolored image of size $m \times n$ ($n \geq m$), the complexity of the max-flow problem is $O(n^{8/3} \log n)$ and the complexity of the min-cost max-flow problem is $O(n^3 \log n)$ [3].

4 Polynomial Case

The reconstruction of bicolored images is a very challenging task in that several authors [21,10,11,7] studied the reconstruction of subproblems.

Fortunately, in practical applications, the projection data are often bounded. In this way, we will study another particular case of bicolored images by assuming that all the projections are bounded i.e., $h_i^a \leq c, h_i^b \leq c, v_j^a \leq c, v_j^b \leq c$ for a given integer c . We also assume that in each line there is at least a colored cell i.e., $h_i^a + h_i^b \geq 1, v_j^a + v_j^b \geq 1$. We denote by C_i the set of columns intersecting row i in a colored cell and by R_j the set of rows intersecting column j in a colored cell. We propose the algorithm A-Bounded. It starts with computing a 2-colored image respecting the projections of 'a' and 'b' with conflict, i.e. some cells can be colored by both colors. At each step, if possible, an interchange operation is carried out to reduce the number of conflicts.

Algorithm A-Bounded

Input: Integral vectors: $H^a \in N^m, V^a \in N^n, H^b \in N^m$ and $V^b \in N^n$.

Output: An approximate bicolored image

Compute x and y solution to $MB(H^a, V^a)$ and $MB(H^b, V^b)$ respectively.

Compute $S = x \oplus y$.

While there is a conflict and an interchange **do**

1. Select a conflict (i, j) in S .
 2. If there is a colored cell (i', j') and the cells (i, j') and (i', j) are uncolored then carry out the interchange $(i, i), (i, j'), (i', j), (i', j')$
- end do**
3. else solve by enumeration the reconstruction problem

Proposition 1. *The algorithm A-Bounded solves in polynomial the particular instance.*

Proof. On Step 2, the number of conflicts decrease at least by one. We will prove that if the algorithm executes Step 3 then $m \leq 2c(2c - 1)$ and $n \leq 2c(2c - 1)$ (see Figure 2). By Step 1, there is a conflict cell (i, j) and indeed $|R_j| \leq v_j^a + v_j^b - 1 \leq 2c - 1$. The number of colored cells in set R_j is not greater than $2c(2c - 1)$ because a row contains at most $2c$ colored cells and $|R_j| \leq 2c - 1$. Thus $n \leq 2c(2c - 1)$ since each column has at least a colored cell in set R_j , otherwise, there is another interchange to carry out. Suppose that column $l \notin C_i$ does not intersect set R_j in a colored cell. Then there exists a colored cell (k, l) such that the row $k \notin R_j$. Thus the cell (i, l) is not colored because $l \notin C_i$ and the cell (k, j) is also not colored because $k \notin R_j$. So the four cells $(i, j), (i, l), (k, j), (k, l)$ constitute an interchange operation. By the same way, we prove that $m \leq 2c(2c - 1)$.

So the size of the problem to solve in the worst case at Step 3 is independent of m and n .

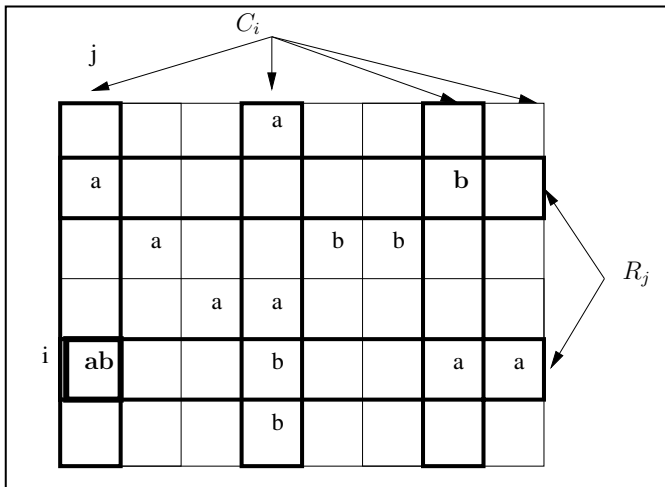


Fig. 2. Bicolored images with bounded projections: step 3

5 Heuristics

We provide the iterative algorithm A-bicolored based on an iterated monocolored min-cost max-flow model to compute an approximate solution to the bicolored images. The aim is to minimize the conflict between the solutions to $MB(H^a, V^a)$ and $MB(H^b, V^b)$. Every iteration consists of solving $MB(H^a, V^a)$ or $MB(H^b, V^b)$ with an objective function depending on the previous iteration. A min-cost max-flow model is used to solve the binary matrix reconstruction problem (see Section 3.2). The costs are chosen in such a way that the new reconstruction has the minimal conflict with the previous one.

Firstly, we compute a solution y^0 to $MB(H^b, V^b)$ by solving the associated max-flow problem. Then, we determine x^1 solution to $MB(H^a, V^a)$ by solving the associated min-cost max-flow problem where the cost matrix $w = y^0$, i.e. a solution that minimizes the conflict with y^0 . Subsequently, x^1 is used as a cost to determine a solution to $MB(H^b, V^b)$. This procedure is repeated until the conflict between the solutions to $MB(H^a, V^a)$ and $MB(H^b, V^b)$ becomes null or constant. As a summary, we can describe the reconstruction algorithm as follows:

Algorithm A-bicolored

Compute y^0 solution to $MB(H^b, V^b)$ by solving the max-flow problem

$$i = 0, f^{-1} = mn + 1, f^0 = mn,$$

While $0 \leq f^i < f^{i-1}$ **do**

Compute x^{i+1} solution to $MB(H^a, V^a)$ minimizing the conflict with y^i .

Compute y^{i+1} solution to $MB(H^b, V^b)$ minimizing the conflict with x^{i+1} .

$f^{i+1} = \text{conf}(x^{i+1}, y^{i+1})$ and $i = i + 1$.

We will give some properties of this polynomial time algorithm.

Claim. i) The conflict is not increasing from iteration to iteration, i.e. $f^{i+1} \leq f^i$.

ii) If $\text{conf}(x^i, y^i) > \text{conf}(x^{i+1}, y^{i+1})$ then $x^{i+1} \neq x^j$ for $j = 1, \dots, i$.

Proof. i) $\text{conf}(x^{i+1}, y^{i+1}) \leq \text{conf}(x^i, y^i)$ because $\text{conf}(x^{i+1}, y^i) \leq \text{conf}(x^i, y^i)$ and $\text{conf}(x^{i+1}, y^{i+1}) \leq \text{conf}(x^{i+1}, y^i)$.

ii) Suppose that $x^{i+1} = x^j$ for some j in the range $1 \leq j \leq i$. By i), $\text{conf}(x^j, y^j) \geq \text{conf}(x^i, y^i)$ and by the algorithm $\text{conf}(x^j, y) \geq \text{conf}(x^j, y^j)$ for all solutions y to $MB(H^b, V^b)$. Hence $\text{conf}(x^j, y) \geq \text{conf}(x^j, y^j) \geq \text{conf}(x^i, y^i) > \text{conf}(x^{i+1}, y^{i+1})$ for all y . In particular for $y = y^{i+1}$, we get $\text{conf}(x^j, y^{i+1}) > \text{conf}(x^{i+1}, y^{i+1})$, a contradiction since $x^{i+1} = x^j$. \square

When the algorithm terminates, the conflict is either null or positive. On the former $S = x^i \oplus y^i$ is a bicolored images satisfying the projections. On the later, an approximate image is computed by arbitrary assigning color 'a' to the half of cells with a conflict and color 'b' to the other half. There is a conflict in a cell if it has value 1 on both matrices x^i and y^i .

6 Results

The main criterion to evaluate the performance of our algorithm is the ability to reconstruct bicolored images. We will try to describe the properties of the heuristic by considering the number of conflicts in the approximate solution.

To test our algorithm, we have used two sets of images. The first set consists of random images of various sizes. The second one consists of some squares shaped images. For each problem, the algorithm either converges to an equivalent image or provides an approximate image. The min-cost max-flow models used by the algorithm are solved by the *CS2* network flow library [16]. Using a Pentium M PC with 2.00 GHz CPU, the run times ranged between 0.5s and 2s for all instances.

6.1 Random Images

We have implemented a program that randomly generates bicolored images. Each pixel has a uniform probability to be colored and if it is colored it has also a uniform probability to be colored by color 'a'. For the first set, the algorithm converges to an equivalent image for all the tested instances and for the most of them it converges rapidly. The second row of Table 1 shows the number of the test instances for each size of images. The third row gives the number of instances where the algorithm converges in two iterations. For the other instances, the algorithm converges in only one iteration. For example, for the images of size (10, 10): Among 235353 test instances, in only 6 instances, the algorithm converges in the second iteration. So for the random images, if a set of cells respecting the projections of 'a' is selected, another set from the remaining ones respecting the projections of 'b' can also be selected. One possible explanation of this fact is that the random images often contain several interchange operations and the solution is not unique.





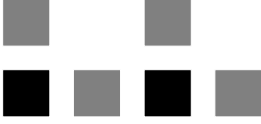

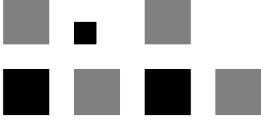
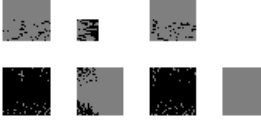
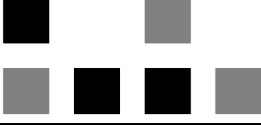

Table 1. Reconstruction of random images

Size	(5,5)	(5,10)	(10,10)	(10,15)	(20,20)	(50,50)	(100,100)	(160,160)
N. ins.	56000	127470	235353	200000	600000	20000	2000	2000
2sd iter.	7	9	6	1	0	0	0	0

6.2 Squares Shaped Images

For the second set of images, the algorithm converges to an equivalent images in two iterations. Table 2 indicates the conflict between the matrices after the first and the second iterations. We note that whenever the test instance has no interchange operation (cases (c) and (d)), the algorithm exactly reconstructs it. Notice that the higher the number of interchange operations, the smaller the conflict in the first iteration becomes. For the reconstructed image (a), we find both colors in the four blocks. That is because the blocks constitute interchange operations.

Table 2. Reconstruction of square shaped images

	Size	Test image	Reconstructed image	$conf(x^1, y^1)$	$conf(x^2, y^2)$
(a)	(80,80)			0	0
(c)	(80,80)			327	0
(d)	(175,80)			114	0
(e)	(175,80)			91	0
(f)	(175,80)			67	0

7 Conclusion

In this paper, we have studied the complexity of reconstructing bicolored images. We have solved a polynomial particular case and provided an iterative algorithm to approximate bicolored images from projections. For evaluation, we have considered two types of images: square shaped and random. The results show that the algorithm gives equivalent images in few iterations.

In general, there is not a unique solution to the problem of reconstructing bicolored images from orthogonal projections. One way to resolve this ambiguity is to increase the number of projections. We are working in extending the algorithm A-bicolored to consider other directions such as diagonal sums and anti-diagonal sums.

References

1. Anstee, R.P.: The network flows approach for matrices with given row and column sums. *Discrete Math.* 44, 125–138 (1983)
2. Barucci, E., Del Lungo, A., Nivat, M., Pinzani, R.: X-rays characterizing some classes of discrete sets. *Linear Algebra and its Applications* 339, 3–21 (2001)

3. Batenburg, K.J.: Network flow algorithms for discrete tomography. In: Herman, G., Kuba, A. (eds.) *Advances in Discrete Tomography and its Applications*, pp. 175–205. Birkhäuser, Boston (2007)
4. Batenburg, K.J.: An evolutionary algorithm for discrete tomography. *Discrete Applied Mathematics* 151, 36–54 (2005)
5. Brunetti, S., Costa, M.C., Frosini, A., Jarray, F., Picouleau, C.: Reconstruction of binary matrices under adjacency constraints. In: Herman, G., Kuba, A. (eds.) *Advances in Discrete Tomography and its Applications*, Boston, pp. 125–150 (2007)
6. Baumann, J., Kiss, Z., Krimmel, S., Kauba, A., Nagy, A., Rodek, L., Schillinger, S., Stephan, J.: Discrete tomography methods for nondestructive testing. In: Herman, G., Kuba, A. (eds.) *Advances in Discrete Tomography and its Applications*, Boston, pp. 303–331 (2007)
7. Brocchi, S., Frosini, A., Rinaldi, S.: Solving some instances of the two color problem. In: Brlek, S., Reutenauer, C., Provençal, X. (eds.) *DGCI 2009. LNCS*, vol. 5810, pp. 505–516. Springer, Heidelberg (2009)
8. Chrobak, M., Dürr, C.: Reconstructing Polyatomic Structures from X-Rays: NP Completeness proof for three Atoms. *Theoretical computer Science* 259(1), 1–98 (2001)
9. Chrobak, M., Couperus, P., Dürr, C., Woeginger, G.: A note on tiling under tomographic constraints. *Theoretical computer Science* 290, 2125–2136 (2003)
10. Costa, M.C., De Werra, D., Picouleau, C.: Using graphs for some discrete tomography problems. *Discrete Applied Mathematics* 154, 35–46 (2006)
11. Costa, M.C., de Werra, D., Picouleau, C., Schindld, D.: *Discrete Applied Mathematics* 148, 240–245 (2005)
12. Dürr, C., Guinez, F., Matamala, M.: Reconstructing 3-colored grids from horizontal and vertical projections is NP-hard, arXiv:0904.3169v1 (2009)
13. Gale, D.: A theorem on flows in networks. *Pacific J. Math.* 7, 1073–1082 (1957)
14. Gardner, R.J., Gritzmann, P., Prangenberg, D.: On the computational complexity of reconstructing lattice sets from their X-rays. *Discrete Mathematics* 202, 45–71 (1999)
15. Gardner, R.J., Gritzmann, P., Prangenberg, D.: On the computational complexity of determining polyatomic structures by X-rays. *Theoretical Computer Science* 233, 91–106 (2000)
16. Goldberg, A.V.: An efficient implementation of a scaling minimum-cost flow algorithm. *Journal of Algorithms* 22, 1–29 (1997)
17. Goldberg, A.V., Rao, S.: Beyond the flow decomposition barrier. *Journal ACM* 45, 783–797 (1998)
18. Hall, P.: A model for learning human vascular anatomy. *DIMACS Serie in Discrete Mathematical Problems with Medical Applications* 55, 11–27 (2000)
19. Herman, G.T., Kuba, A.: *Advances in Discrete Tomography and its Applications*. Birkhäuser, Boston (2007)
20. Jarray, F.: Solving problems of discrete tomography. Applications in workforce scheduling, Ph.D. Thesis, University of CNAM, Paris (2004)
21. Jarray, F.: Workforce scheduling and table coloring. In: *Proceedings ROADEF 2005*, Tours, pp. 92–100 (2004)
22. Jarray, F.: A lagrangean approach to reconstruct bicolored images from discrete orthogonal projections. *Pure mathematics and applications (Linear algebra and computer science)* 20(1), 17–25 (2010)
23. 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)

24. Onnasch, D.G.W., Prause, G.P.M.: Heart Chamber Reconstruction from Biplane Angiography. In: Herman, G., Kuba, A. (eds.) *Discrete Tomography: Foundations, Algorithms and Applications*, pp. 385–403. Birkhäuser, Boston (1999)
25. Ryser, H.J.: Combinatorial properties of matrices of zeros and ones. *Canad. J. Math.* 9, 371–377 (1957)
26. Sachs, J.R.J., Sauer, K.: 3D Reconstruction from sparse Radiographic Data. In: Herman, G., Kuba, A. (eds.) *Discrete Tomography: Foundations, Algorithms and Applications*, pp. 363–383. Birkhäuser, Boston (1999)
27. Salzberg, P.M., Rivera-vega, P.I., Rodriguez, A.: Network flow model for binary tomography on lattices. *Internal journal imaging system technology* 9, 145–154 (1998)
28. Slump, C.H., et al.: A network flow approach to reconstruction of the left ventricle from two projections. *Comput. Gr. Im. Proc.* 18, 18–36 (1982)
29. Wang, B., Zhang, F.: On the precise Number of $(0,1)$ -Matrices in $\mathcal{U}(R, S)$. *Discrete Mathematics* 187, 211–220 (1998)

Discrete Q-Convex Sets Reconstruction from Discrete Point X-Rays*

Fatma Abdmouleh, Alain Daurat, and Mohamed Tajine

LSIIT CNRS UMR 7005, Strasbourg University
Pole API Boulevard Sebastien Brant 67412 Illkirch-Graffenstaden, France

Abstract. The problem of reconstructing sets from their point X-rays is considered. We study the problem for Q-convex sets which are sets having special convexity properties. These properties allow the reconstruction with few projections. In this paper we introduce the filling operations adapted to the considered context and we provide an algorithm for reconstructing Q-convex sets from their point X-rays for two source points. The reconstruction of Q-convex sets would be an intermediate step for reconstructing convex sets from their point X-rays.

Keywords: convexity, point X-rays, discrete sets, discrete tomography, filling operations, Q-convex sets.

1 Introduction

Tomography deals with the inverse problem of reconstructing an object from its projections. From a mathematical point of view, this problem is equivalent to determining an unknown function, from its integrals (continuous case) or its weighted sums (discrete case) over subspaces of its domain [14]. In discrete tomography, the range of this function is a given discrete set.

The particularity of discrete tomography is that only few number of projections are needed to resolve the inverse problem. These projections may be collected from parallel X-rays or from point X-rays. The point X-rays framework can be seen as the generalization of the parallel X-rays framework. Indeed, Parallel X-rays is an approximation of a point X-rays where the light source point is placed at an infinite distance from the object.

In this work, we study the reconstruction of discrete objects from their discrete projections. Many results are available for discrete parallel X-rays. R. J. Gardner and P. Gritzmann showed in [12] that any set of seven mutually non parallel lattice directions determine convex subsets of \mathbb{Z}^2 . In [13], the authors show that it is possible to find four projections that will uniquely determine all planar convex bodies. In [5], the authors prove the stability of the reconstruction problem for convex sets. Furthermore, in paper [2] a polynomial-time algorithm

* This work was supported by the Agence Nationale de la Recherche through contract ANR-2010-BLAN-0205-01.

for reconstructing a discrete set with special connectivity and convexity properties in directions $(1, 0)$, $(0, 1)$, and $(1, 1)$ is provided. A similar idea is employed by A. Daurat in his Ph.D thesis [8] where he defines a new class of subsets of \mathbb{Z}^2 called ‘Q-convex’ and studies the reconstruction problem for these subsets. Several works have been devoted to the study of this class of sets [5,6,9,7,10]. S. Brunetti and A. Daurat present in [6] and [9] a study of the ‘Q-convex’ sets. They also provide two random generator for these subsets in [7]. In [4], they provide a polynomial algorithm that enables the reconstruction of this kind of sets for two directions and that can be used for reconstructing any convex subsets of \mathbb{Z}^2 for some suitable four directions or for any seven mutually nonparallel directions.

However, there are less results for the discrete point X-rays. An initialization of studies of discrete point X-rays is made by P. Dulio, R. J. Gardner and C. Peri in [11] where they show that the problem is solved for some sets of four collinear source points and for any set of at least seven collinear points where six collinear points are generally not enough. They also show that for a set of four source points, no three of them are collinear, convex lattice sets not meeting any line joining two of these points are not determined by discrete point X-rays from this set.

In the present paper, we study the reconstruction problem for a similar class of sets as the class introduced by A. Daurat, the Q-convex sets for point sources. A more detailed study of this class of sets is realized in [1].

The second section of this paper is dedicated to the introduction of continuous and discrete Q-convex sets for point sources. The third section presents adapted filling operation that we use in an algorithm for solving the reconstruction problem discrete Q-convex sets for a couple of point sources. In the last section, we apply the algorithm on a set for which we compute the projections. These projections are then our data for the reconstruction task.

2 Definition and Notations

In this section, we introduce the notions that constitute the basic ingredients of our work.

2.1 Classical Definitions and Notations

Definition 1. *A set $E \subset \mathbb{R}^2$ is convex (or \mathbb{R} -convex) if for every $A, B \in E$ we have $[A, B] \subseteq E$ where $[A, B] = \{\lambda A + (1 - \lambda)B \mid 0 \leq \lambda \leq 1\}$ is called the line segment between A and B .*

In all the following, \mathbb{S} denotes \mathbb{R} or \mathbb{Z} .

Let E be a subset of \mathbb{R}^2 and A and B be two distinct points of \mathbb{R}^2 . In this paper, we use the following notations:

- $\mathcal{C}(\mathbb{R}^2)$ is the set of all convex subsets of \mathbb{R}^2 .
- $\mathcal{C}_E(\mathbb{R}^2, E) = \{E' \in \mathcal{C}(\mathbb{R}^2) \mid E \subseteq E'\}$ is the set of all convex subsets of \mathbb{R}^2 containing the set E .

- $CH(E) = \bigcap_{E' \in \mathcal{C}_E(\mathbb{R}^2, E)} E'$ is the convex hull of E .
- (AB) is the straight line joining A and B .
- A ray or a half-line $R_{S,\theta}$ from a point $S = (x_0, y_0)$ in the direction $\mathbf{u}_\theta = (u_1, u_2)$ where $\sqrt{u_1^2 + u_2^2} = 1$, and $\cos \theta = u_1$ and $\sin \theta = u_2$ can be defined in different ways:

$$\begin{aligned} R_{S,\theta} &= \{(x, y) \in \mathbb{R}^2 \mid u_2(x - x_0) - u_1(y - y_0) = 0 \text{ and } x \geq x_0\}; \\ &= \{(x_0, y_0) + \lambda \mathbf{u}_\theta \mid \lambda \geq 0\}; \\ &= \{M \in \mathbb{R}^2 \mid \widehat{PSM} = \theta\}; \end{aligned}$$

where \widehat{PSM} denotes the angle between (SP) and (SM) with $P = S + (1, 0)$ (see Fig. 1). In all the following, the angle \widehat{PSM} is denoted θ_{SM} .

We also introduce:

- For a point $S \in \mathbb{R}^2$, we define the set of all the angles of all the rays issuing from S and passing through all the points of \mathbb{S}^2 relatively to the horizontal line passing through $P = S + (1, 0)$:

$$\mathcal{A}(S, \mathbb{S}^2) = \{\theta_{SM} \mid M \in \mathbb{S}^2\}.$$

- If E is a finite subset, then $|E|$ is the cardinality of E indicating the number of elements of E .
- Let A be a set. We denote by $\mathfrak{P}(A)$ the powerset of A ($\mathfrak{P}(A) = \{B \mid B \subseteq A\}$).

Remark 1. Let $S \in \mathbb{R}^2$:

- $\mathcal{A}(S, \mathbb{R}^2) = [0, 2\pi[$.
- $\mathcal{A}(S, \mathbb{Z}^2)$ is an infinite countable set.
- $\overline{\mathcal{A}(S, \mathbb{Z}^2)} = [0, 2\pi]$; where \overline{F} is the closure of the set F in \mathbb{R} relatively to the usual topology.

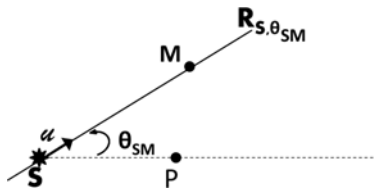


Fig. 1. The ray R is defined by the initial point S and the angle θ_{SM}

For the discrete sets, convexity can be defined as follows :

Definition 2. Let $D \subset \mathbb{Z}^2$. D is \mathbb{Z} -convex if $D = CH(D) \cap \mathbb{Z}^2$.

Let $E \subseteq \mathbb{R}^2$ and a point S in \mathbb{R}^2 . We define the projection of E from the source point S denoted $X_S(E, \cdot): \mathbb{R} \mapsto \mathbb{R}$ by:

$$X_S(E, \theta) = \int_0^{+\infty} \chi_{E \cap R_{S,\theta}}(S + t\mathbf{u}_{\theta_{SM}}) dt.$$

Where $\mathbf{u}_{\theta_{SM}} = (\cos \theta, \sin \theta)$ and

$$\chi_E(x) = \begin{cases} 1 & \text{if } x \in E \\ 0 & \text{otherwise.} \end{cases}$$

Then, $X_S(E, \theta) = \mu(E \cap R_{S,\theta})$ where μ is the usual measurement on \mathbb{R} .

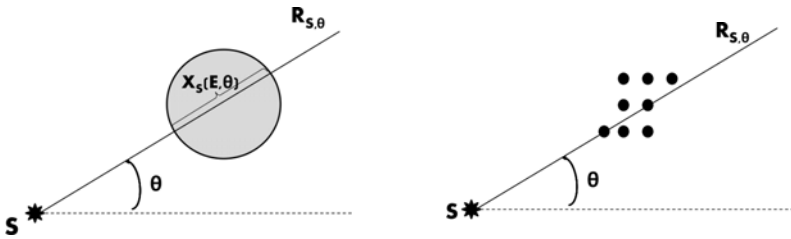


Fig. 2. Continuous (left) and discrete(right) point X-rays

Now, let us consider a source point S and a finite subset $D \subset \mathbb{Z}^2$. We have a finite number of rays issuing from S and passing through all the points of D and each of these rays passes through a finite number of points of D . The projection of D from the source point S is the function $X_S(D, \cdot): \mathbb{R} \mapsto \mathbb{N}$ such that:

$$X_S(D, \theta) = |R_{S,\theta} \cap D|.$$

Definition 3. Let $E \subset \mathbb{S}^2$. The support of E for the source point S is the set:

$$Supp_{\mathbb{S}}(E, S) = \{\theta \in \mathcal{A}(S, \mathbb{S}^2) \mid X_S(E, \theta) \neq 0\}.$$

For the reconstructing problem in both the continuous and discrete cases, we aim to reconstruct the subset E of \mathbb{R}^2 or the finite subset D of \mathbb{Z}^2 having the position of the source point and a set of angles and their projections (which is infinite for the continuous case and finite for the discrete one). Only the rays $R_{S,\theta}$ such that $\theta \in Supp_{\mathbb{S}}(E, S)$ are considered.

2.2 Q-Convexity

In this part, we introduce both continuous and discrete Q-convex sets for two source points. This notion was first introduced by A. Daurat [8] for discrete parallel rays.

Considering a ray $R_{S,\theta}$ from a point S , we define:

- $\mathcal{L}(R_{S,\theta}) = \{M \in \mathbb{S}^2 \mid 0 \leq \theta_{SM} - \theta \leq \pi\}$ the set of points that are on the left of the straight line containing $R_{S,\theta}$;
- $\mathcal{R}(R_{S,\theta}) = \{M \in \mathbb{S}^2 \mid -2\pi \leq \theta_{SM} - \theta \leq \pi\}$ the set of points that are on the right of the straight line containing $R_{S,\theta}$.

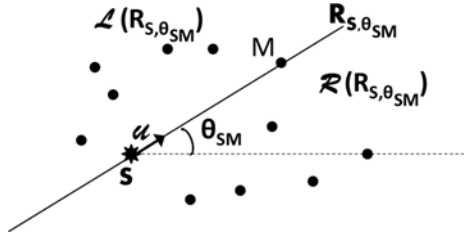


Fig. 3. Linear separation of the plane by the straight line containing the ray $R_{S,\theta}$

We consider two distinct source points S and $S' \in \mathbb{R}^2$, a set $E \subset \mathbb{S}^2$, two rays $R_{S,\theta_{SM}}$ and $R_{S',\theta'_{S'M}}$ such that $R_{S,\theta_{SM}} \cap R_{S',\theta'_{S'M}} = \{M\}$. This intersection defines the following four zones (called quadrants):

$$\begin{aligned} Z^0_{\{S,S'\}}(M) &= \mathcal{R}(R_{S,\theta_{SM}}) \cap \mathcal{L}(R_{S',\theta'_{S'M}}), \\ Z^1_{\{S,S'\}}(M) &= \mathcal{R}(R_{S,\theta_{SM}}) \cap \mathcal{R}(R_{S',\theta'_{S'M}}), \\ Z^2_{\{S,S'\}}(M) &= \mathcal{L}(R_{S,\theta_{SM}}) \cap \mathcal{R}(R_{S',\theta'_{S'M}}), \\ Z^3_{\{S,S'\}}(M) &= \mathcal{L}(R_{S,\theta_{SM}}) \cap \mathcal{L}(R_{S',\theta'_{S'M}}). \end{aligned}$$

Definition 4. A set $E \subset \mathbb{S}^2$ is \mathbb{S} -Q-convex (quadrant-convex) for two source points S and S' if, for all $M \in \mathbb{S}^2$, we have :

$$\forall t \in \{0, 1, 2, 3\}, Z^t_{\{S,S'\}}(M) \cap E \neq \emptyset \implies M \in E$$

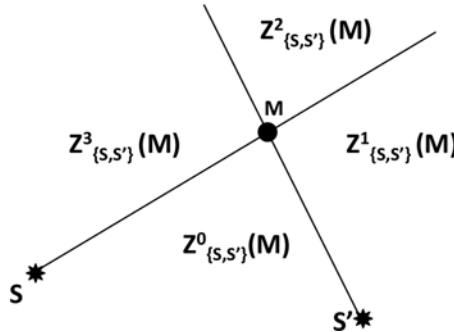


Fig. 4. Four zones resulting from the intersection between two rays in a point M

Definition 5. Let \mathcal{P} be a set of at least two source points. We say that a set $E \subset \mathbb{S}^2$ is \mathbb{S} -Q-convex for \mathcal{P} if it is \mathbb{S} -Q-convex for any two points of \mathcal{P} .

We denote the class of sets which are \mathbb{S} -Q-convex for a set of source points \mathcal{P} by $\mathcal{QC}_{\mathbb{S}}(\mathcal{P})$.

The reader might want to refer to [1] where we present a deeper study of the Q-convex sets for point sources.

3 Reconstruction Algorithm for Two Source Points

This section is dedicated to the reconstruction problem of the Q-convex sets introduced in the last section.

3.1 Problem Presentation

Let us introduce the reconstruction problem for a set $\mathcal{P} = \{S, S'\}$ of source points. The aim is to reconstruct a finite \mathbb{Z} -Q-convex set D from its projections relatively to S and S' . We denote by j the index of the ray R_{S, θ_j} issued from S and by i the index of the ray R_{S', θ'_i} issued from S' .

Then, the data available for the reconstruction is the position of the two points S and S' , their respective supports $Supp_{\mathbb{Z}}(D, S) = \{\theta_1, \dots, \theta_j, \dots, \theta_n\}$ where angles are sorted in an ascending order and $Supp_{\mathbb{Z}}(D, S') = \{\theta'_1, \dots, \theta'_i, \dots, \theta'_m\}$ where angles are sorted in a descending order, and two vectors $\mathbb{P}_S(D) = (s_1, \dots, s_j, \dots, s_n)$ and $\mathbb{P}_{S'}(D) = (s'_1, \dots, s'_i, \dots, s'_m)$ of nonnegative integers where s_j is the number of points of D along j^{th} ray from S and s'_i is the number of points of D along the i^{th} ray from S' .

Hence, the reconstructed set $D \in \mathcal{QC}_{\mathbb{Z}}(\{S, S'\})$ should satisfy $X_S(D, \theta_j) = s_j$ and $X_{S'}(D, \theta'_i) = s'_i$ for all $j \in \llbracket 1, n \rrbracket$ and $i \in \llbracket 1, m \rrbracket$.

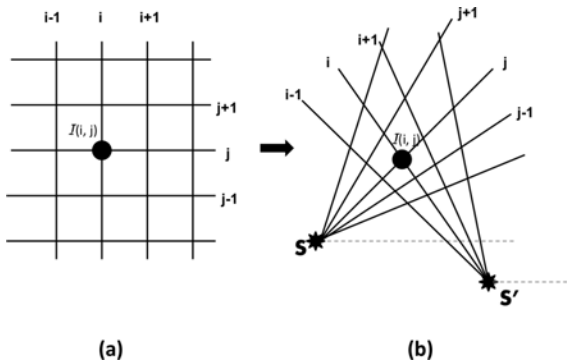


Fig. 5. (a) Grill obtained by parallel rays - (b) Deformed grill obtained by rays from two point sources

We denote by $\mathcal{I}(i, j)$, the point such that $\mathcal{I}(i, j) = R_{S, \theta_j} \cap R_{S', \theta_i}$. Note that $\mathcal{I}(i, j)$ is not always in \mathbb{Z}^2 . Hence, we obtain a deformed grill as shown in Fig 5. We define:

- $\Delta = \{ \mathcal{I}(i, j) \in \mathbb{Z}^2 : 1 \leq i \leq m, 1 \leq j \leq n \}$;
- $A(S, \mathbb{P}_S(D)) = \sum_{j=1}^n s_j$.

A solution of the problem D should necessarily satisfy the following conditions:

1. $D \subseteq \Delta$ (convex constraint).
2. $|D| = A(S, \mathbb{P}_S(D)) = A(S', \mathbb{P}_{S'}(D))$ (conservation constraint).
3. Given the j^{th} ray R_{S, θ_j} from S and two points $M_1, M_2 \in D \cap R_{S, \theta_j}$. All points of $R_{S, \theta_j} \cap \mathbb{Z}^2$ lying between M_1 and M_2 are point of D and there exists a ray from S' passing through each of these points. The same property is satisfied for the rays from S' (see Fig 6) (closure of the support constraint).

Hence, if these conditions are not fulfilled, we can directly deduce that there is no solution to the considered problem.

In order to reconstruct the Q-convexes, we will consider a lower bound called “kernel” α and an upper bound called “feedstock” β of the solutions D such that we have:

$$\alpha \subseteq D \subseteq \beta.$$

Then, for any i and j we have:

$$\begin{aligned} |R_{S, \theta_j} \cap \alpha| &\leq s_j; & |R_{S', \theta'_i} \cap \alpha| &\leq s'_i; \\ |R_{S, \theta_j} \cap \beta| &\geq s_j; & |R_{S', \theta'_i} \cap \beta| &\geq s'_i. \end{aligned}$$

The algorithm aims to approach D by increasing α and decreasing β .

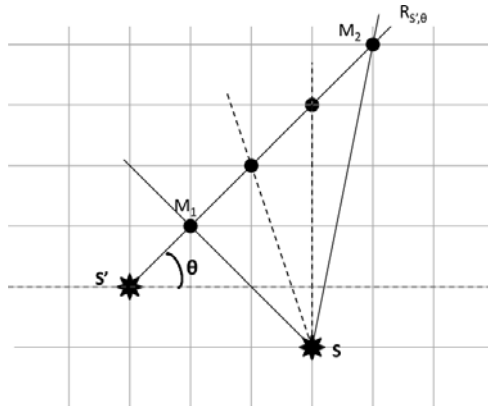


Fig. 6. All points between M_1 and M_2 are in D and there exists a ray from S passing through each of them

3.2 Filling Operations

In this subsection, we consider two sets α and β of \mathbb{Z}^2 such that for all solutions $D \in \mathcal{QC}_{\mathbb{Z}}(S, S')$ having the projections (s'_i) and (s_j) we have:

$$\alpha \subseteq D \subseteq \beta.$$

A filling operation gives new subsets α' and β' such that for $D \in \mathcal{QC}_{\mathbb{Z}}(S, S')$ having the projections $(s_j)_{j \in \llbracket 1, n \rrbracket}$ and $(s'_i)_{i \in \llbracket 1, m \rrbracket}$ we have:

$$\alpha \subseteq \alpha' \subseteq D \subseteq \beta' \subseteq \beta.$$

Then, we have:

$$\begin{aligned} |R_{S, \theta_j} \cap \alpha| &\leq |R_{S, \theta_j} \cap \alpha'| \leq s_j; & |R_{S', \theta'_i} \cap \alpha| &\leq |R_{S', \theta'_i} \cap \alpha'| \leq s'_i; \\ |R_{S, \theta_j} \cap \beta| &\geq |R_{S, \theta_j} \cap \beta'| \geq s_j; & |R_{S', \theta'_i} \cap \beta| &\geq |R_{S', \theta'_i} \cap \beta'| \geq s'_i. \end{aligned}$$

Notations. We introduce some notations that will be of use in the sequel.

Let $A \subseteq \mathbb{Z}^2$. We denote by $A_i(S')$ the intersection between A and the i^{th} ray from S' and by $A^j(S)$ the intersection between A and the j^{th} ray from S .

For a given ray R_{S, θ_j} , we introduce the following notations:

- $r(A^j(S)) = \max \{i \mid \mathcal{I}(i, j) \in A^j(S)\}$, $l(A^j(S)) = \min \{i \mid \mathcal{I}(i, j) \in A^j(S)\}$;
- $\sigma_r^A(i, s_j) = i_1 - i$ where i_1 is such that $|(\bigcup_{i \leq i' \leq i_1} R_{S, \theta_{i'}}) \cap A^j(S)| = s_j$;
- $\sigma_l^A(i, s_j) = i - i_2$ where i_2 is such that $|(\bigcup_{i_2 \leq i' \leq i} R_{S, \theta_{i'}}) \cap A^j(S)| = s_j$.

$\sigma_r^A(i, s_j)$ (respectively $\sigma_l^A(i, s_j)$) gives the number of rays separating the point $\mathcal{I}(i, j)$ and s_j^{th} points of \mathbb{Z}^2 on the right (respectively on the left) of $\mathcal{I}(i, j)$.

Operations on Rays. We first present the four operations \oplus , \otimes , \ominus and \odot that treat each ray separately. These operations were first introduced for the parallel X-ray case in [3] for the horizontal and vertical directions and adapted in [4] for any direction.

- If $\alpha^j(S) \neq \emptyset$, then $\oplus \alpha^j(S) = \{\mathcal{I}(i, j) \in \mathbb{Z}^2 \mid l(\alpha^j(S)) \leq i \leq r(\alpha^j(S))\}$ (see Fig 7(a)).
- $\otimes \alpha^j(S) = \{\mathcal{I}(i, j) \in \mathbb{Z}^2 \mid r(\beta^j(S)) - \sigma_l^\beta(r(\beta^j(S)), s_j) \leq i \leq l(\beta^j(S)) + \sigma_r^\beta(l(\beta^j(S)), s_j)\}$ (see Fig 7(b)). This operation adds no points to $\alpha^j(S)$ if $r(\beta^j(S)) - \sigma_l(r(\beta^j(S)), s_j) > l(\beta^j(S)) + \sigma_r(l(\beta^j(S)), s_j)$.
- • If $\alpha^j(S) \neq \emptyset$, $\mathcal{I}(i', j) \notin \beta^j(S)$ with $i' \leq l(\alpha^j(S))$, then:

$$\ominus \beta^j(S) = \{\mathcal{I}(i, j) \in \beta^j(S) \mid i > i'\}.$$

- If $\alpha^j(S) \neq \emptyset$, $\mathcal{I}(i', j) \notin \beta^j(S)$ with $i' \geq r(\alpha^j(S))$, then:

$$\odot \beta^j(S) = \{\mathcal{I}(i, j) \in \beta^j(S) \mid i < i'\} \text{ (see Fig 7(c)).}$$

– If $\alpha^j(S) \neq \emptyset$, then:

$$\odot \beta^j(S) = \{ \mathcal{I}(i, j) \in \beta^j(S) \mid r(\alpha^j(S)) - \sigma_i^\alpha(r(\alpha^j(S)), s_j) < i < l(\alpha_j) + \sigma_r^\alpha(l(\alpha_j), s_j) \}.$$

(see Fig 7(d))

Remark 2. For a given ray R_{S, θ_j} , if $\mathcal{I}(l(\beta^j(S)), j) = (x_1, y_1)$ and $\mathcal{I}(r(\beta^j(S)), j) = (x_2, y_2)$. The number of points in $R_{S, \theta_j} \cap \mathbb{Z}^2$ lying between $\mathcal{I}(l(\beta^j(S)), j)$ and $\mathcal{I}(r(\beta^j(S)), j)$ is equal to: $\lfloor \frac{x_2 - x_1}{b} \rfloor$ where $\frac{y_2 - y_1}{x_2 - x_1} = \frac{a}{b}$ with $\gcd(a, b) = 1$. So, if we have $\lfloor \frac{x_2 - x_1}{b} \rfloor > 2s_j$, then operation \otimes adds no points to $\alpha^j(S)$.

We also define two other operations \odot' and \odot'' that are called coherence operations on $\beta^j(S)$:

- If $s_j = 0$, then $\odot' \beta^j(S) = \emptyset$.
- If $\mathcal{I}(i', j), \mathcal{I}(i'', j) \notin \beta$ and there is less than s_j points of \mathbb{Z}^2 separating $\mathcal{I}(i', j)$ and $\mathcal{I}(i'', j)$, then $\odot'' \beta^j(S) = \{ \mathcal{I}(i, j) \in \beta^j(S) \mid i < i' \text{ or } i > i'' \}$. This operation eliminates all sequences of $\beta^j(S)$ that are less than s_j (see Fig 7(e)).

Remark 3. For any $\odot_1 \in \{ \oplus, \otimes \}$, $\odot_2 \in \{ \ominus, \odot, \odot', \odot'' \}$, we have:

$$\begin{aligned} \alpha' &= (\alpha \setminus \alpha^j(S)) \cup \odot_1 \alpha^j(S), \\ \beta' &= (\beta \setminus \beta^j(S)) \cup \odot_1 \beta^j(S). \end{aligned}$$

The same operations are defined as well for $\alpha_i(S')$ and $\beta_i(S')$.

Global Operations. The filling operations described below only depend on the projection vectors $\mathbb{P}_{S'}(D)$ and $\mathbb{P}_S(D)$. To define it, we first introduce four partial sums :

$$\begin{aligned} \mathcal{S}_0(\mathcal{I}(i, j)) &= \mathcal{S}_0(i) = \mathbb{P}_{S'}(\mathcal{L}(R_{S', \theta'_i})), \quad \mathcal{S}_2(\mathcal{I}(i, j)) = \mathcal{S}_2(i) = \mathbb{P}_{S'}(\mathcal{R}(R_{S', \theta'_i})), \\ \mathcal{S}_1(\mathcal{I}(i, j)) &= \mathcal{S}_1(j) = \mathbb{P}_S(\mathcal{R}(R_{S, \theta_j})), \quad \mathcal{S}_3(\mathcal{I}(i, j)) = \mathcal{S}_3(j) = \mathbb{P}_S(\mathcal{L}(R_{S, \theta_j})). \end{aligned} \quad (2)$$

The following property puts emphasis on the relation between the zones and the partial sums.

Proposition 1. *If $\mathcal{S}_t(M) + \mathcal{S}_{t+1}(M) > |D|$, then $Z^t(M) \cap D \neq \emptyset$. If $\mathcal{S}_t(M) + \mathcal{S}_{t+1}(M) \geq |D|$ and $Z^t(M) \cap D = \emptyset$, then $D \subset Z^{t-1}(M) \cap Z^{t+1}(M)$.*

Proof. Let us assume that $Z^t(M) \cap D = \emptyset$. In this case, we necessarily have $\mathcal{S}_t(M) + \mathcal{S}_{t+1}(M) = |D|$. But this would be impossible if $\mathcal{S}_t(M) + \mathcal{S}_{t+1}(M) > |D|$ and we naturally have $D = D \cap (Z^{t-1}(M) \cap Z^{t+1}(M))$ if $\mathcal{S}_t(M) + \mathcal{S}_{t+1}(M) = |D|$.

We now come to the global filling operations \oplus' and \ominus' .

Let $M = \mathcal{I}(i_M, j_M)$ be such that, for all t , we have: $Z^t(M) \cap \alpha \neq \emptyset$ and $\mathcal{S}_t(M) + \mathcal{S}_{t+1}(M) > \mathbf{A}(S, \mathbb{P}_S(D))$.

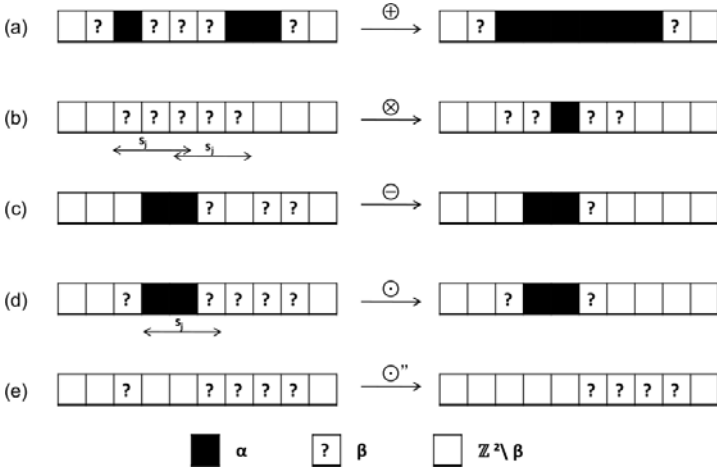


Fig. 7. Filling operations on each ray

- If $M \in \mathbb{Z}^2$, then $\oplus' \alpha = \alpha \cup \{M\}$.
- If $M \notin \mathbb{Z}^2$, then :
 - $\ominus' \beta_{i_M}(S') = \{j_M - \sigma_r(j_M, s'_{i_M}) < j < j_M + \sigma_l(j_M, s'_{i_M})\}$
 - $\ominus' \beta^{j_M}(S) = \{i_M - \sigma_r(i_M, s_{j_M}) < i < i_M + \sigma_l(i_M, s_{j_M})\}$

Remark 4. Applying operations \oplus' to α and \ominus' to β gives:

$$\alpha' = \oplus' \alpha$$

and

$$\beta = (\beta \setminus (\beta_i(S') \cup \beta^j(S))) \cup \{\ominus' \beta_{i_M}(S) \cup \ominus' \beta^{j_M}(S)\}.$$

We will now prove that the presented filling operations are valid.

Proposition 2. For any $\odot_1 \in \{\oplus, \otimes, \oplus'\}$, $\odot_2 \in \{\ominus, \odot, \odot', \odot'', \ominus'\}$ we have :

$$\text{For any solution } D, \alpha \subseteq \odot_1 \alpha \subseteq D \subseteq \odot_2 \beta \subseteq \beta.$$

Proof. For all operations other than \oplus' and \ominus' , the convexity is sufficient to prove the proposition. For the \oplus' and \ominus' operations, we consider a solution D such that $\alpha \subseteq D \subseteq \beta$ and a point $M = \mathcal{I}(i_M, j_M)$ such that $Z^t(M) \cap \alpha \neq \emptyset$ or $\mathcal{S}_t(M) + \mathcal{S}_{t+1}(M) > |D|$, for all t .

Proposition 1 says that for all t we have $Z^t(M) \cap D \neq \emptyset$. Then if $M \in \mathbb{Z}^2$, we can deduce by Q-convexity that $M \in D$. Then \oplus' is a valid operation.

Now let us assume that $M \notin \mathbb{Z}^2$. Let $A = \mathcal{I}(i_A, j_M), B = \mathcal{I}(i_B, j_M) \in \mathbb{Z}^2$ be the two points of ray j_M surrounding M such that $i_B < i_M < i_A$. We assume that $s_{j_M} \neq 0$, then there must be a point $N \in D$ such that $N = \mathcal{I}(i_N, j_M) \in D$. If $i_N < i_M$, then we have $i_N \leq i_B$ which means that $N \in Z^0(B)$ and $N \in$

$Z^3(B)$. However, we have $Z^1(B) \cap D \neq \emptyset$ since $Z^1(M) \subset Z^1(B)$ and equally $Z^2(B) \cap D \neq \emptyset$. Since D is Q-convex we have $B \in D$. Likewise, if $i_N > i_M$ we can deduce that $A \in D$. Hence $\{A, B\} \cap D \neq \emptyset$. We can then see that for any point $N = \mathcal{I}(i_N, j_M)$ of D we have $i_B - \sigma_r(i_M, s'_{j_M}) \leq i_N \leq i_M \sigma_l(i_A, s'_{j_M})$ which proves that $D \subset \ominus' \beta$.

We recall that the reconstruction problem consists in determining if the set of solutions D such that $\emptyset \subseteq D \subseteq \mathbb{Z}^2$ is empty and in reconstructing a solution of one exists. We can clearly see that for such solution D we have: $\emptyset \subseteq D \subseteq \Delta \subseteq \mathbb{Z}^2$ where Δ is described above. To initialize α , we can choose candidate points. For example, we can fix $U_1 = \mathcal{I}(1, j_1)$ and $U_2 = \mathcal{I}(m, j_n)$ from Δ . U_1 and U_2 are called s-bases.

The filling operations are applied on the rays from S and S' and repeated until we have $\alpha \not\subseteq \beta$ or no further changes in α and β . In the first case, we conclude that there is no solution for such α and β . We choose then different s-bases and try again.

The second case can occur for $\alpha = \beta$ and for $\alpha \subset \beta$ where $\beta \setminus \alpha \neq \emptyset$. If we obtain $\alpha = \beta$, then we just have to check that α is Q-convex. The solution to our problem would then be α . If we have $\beta \setminus \alpha \neq \emptyset$, then we need to find a solution by adding to α a set of points from $\beta \setminus \alpha \neq \emptyset$ and verifying if we obtain a solution of the problem.

The algorithm is:

Algorithm 1.

Initialize $\alpha = \{U_1, U_2\}$, $\beta = \Delta$

repeat

 Choose $\odot_1 \in \{\oplus, \otimes, \oplus'\}$, $\odot_2 \in \{\ominus, \odot, \odot', \odot'', \ominus'\}$

for $j = 1$ to n **do**

$\alpha \leftarrow (\alpha \setminus \alpha^j(S)) \cup \odot_1 \alpha^j(S)$

$\beta \leftarrow (\beta \setminus \beta^j(S)) \cup \odot_2 \beta^j(S)$

end for

for $i = 1$ to m **do**

$\alpha \leftarrow (\alpha \setminus \alpha_i(S')) \cup \odot_1 \alpha_i(S')$

$\beta \leftarrow (\beta \setminus \beta_i(S')) \cup \odot_2 \beta_i(S')$

end for

until stability or $\alpha \not\subseteq \beta$

if $\alpha \not\subseteq \beta$ **then**

return "no solution for such α, β "

end if

if $\alpha = \beta$ **then**

return α

end if

find a set $D \in \mathcal{QC}(\mathcal{P})$ solution of the problem verifying $\alpha \subseteq D \subseteq \beta$

4 Example

This section shows an example where we apply the algorithm provided in the previous section. For collecting data, we consider two point sources S and S' and we randomly generate a \mathbb{Z} -Q-convex set for $\{S, S'\}$ D (The reader can refer to [1] for random generation of \mathbb{Z} -Q-convex sets). The set obtained is illustrated in Fig.8. Then, the data we have for the reconstruction task, is:

- The support of D for the source point S is $Supp_{\mathbb{Z}}(D, S) = \{\theta_1, ..\theta_6\}$ with $\mathbb{P}_S(D) = (1, 1, 1, 1, 1, 3)$,
- The support of D for the source point S' is $Supp_{\mathbb{Z}}(D, S') = \{\theta'_1, ..\theta'_6\}$ with $\mathbb{P}_{S'}(D) = (1, 3, 1, 1, 1, 1)$.

The set Δ is illustrated in Fig.9. We initialize the kernel $\alpha = \emptyset$ and the feedstock $\beta = \Delta$. By applying the gobal operation \oplus' on the rays of S and the operation \otimes on the ray R_{S, θ_6} gives new sets α and β illustrated in Fig. 10. The remaining

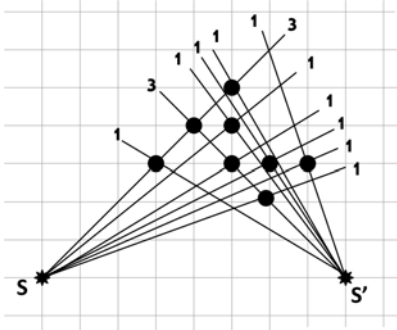


Fig. 8. Q-convex set D

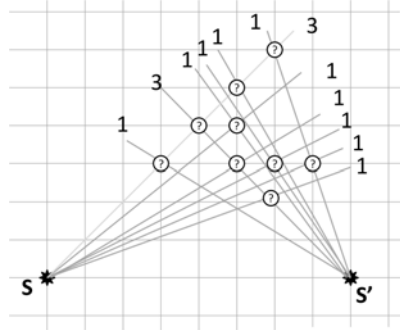


Fig. 9. The set Δ

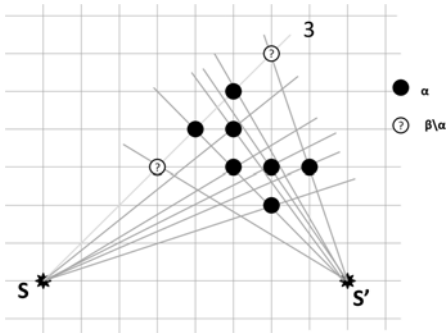


Fig. 10. α and β obtained after filling operations on rays of S

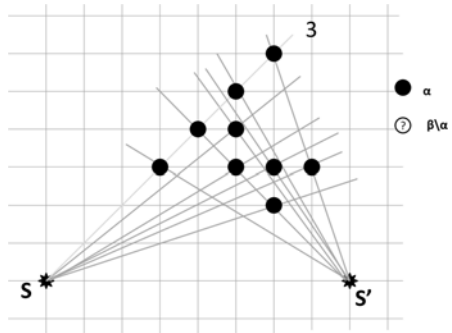


Fig. 11. β obtained after filling operations on rays of S'

undetermined points will be determined after applying the filling operations on the rays of S' . Indeed, the global operation on the rays of S' adds the point $\mathcal{I}(1, 6)$ to α and the operation \ominus applied on the ray R_{S', θ'_6} removes the point $\mathcal{I}(6, 6)$ from β . We then obtain the sets α and β illustrated in Fig. [III](#).

The application of any other filling operation will bring no more changes to the obtained sets α and β and we still have $\alpha \subseteq \beta$. Then, we are in the stability case. Following the algorithm steps, we check that $\alpha = \beta$. Hence the solution of this reconstruction problem is the obtained set α .

5 Conclusion

We have presented in this paper the reconstruction problem for a new class of sets that is the Q-convex sets for point X-rays. In the second section we defined the continuous and discrete Q-convex sets for point X-rays. This may allow us to study more connections between discrete and continuous reconstruction problem. Meanwhile, we dedicated the third section to resolving this problem in the discrete case. We adapted the existing filling operations to our case and used it in an algorithm, showing that the same results obtained on discrete Q-convex sets for parallel X-rays can be obtained with the point rays. We aim to study this algorithm more deeply and to see if it can be optimized for a unique and 'fast' solution of the reconstruction problem of such sets.

In the last section, we illustrated an example for the application of the algorithm where, after considering a Q-convex set, we computed its projection, and starting from these projection we could reconstruct the set. In the example, the algorithm converged immediately. In some other cases, we could have had to choose a solution set using a 2-SAT formula [\[8\]](#). Another perspective that we have is to study and optimize the resolution of such cases.

Acknowledgements. Alain Daurat, co-author of this article, died on June the 25th, 2010. This article is dedicated to his memory.

References

1. Abdmouleh, F., Daurat, A., Tajine, M.: Q-convex sets for point sources (2010), <http://hal.archives-ouvertes.fr/hal-00563126> (preprint)
2. Barucci, E., Brunetti, S., Del Lungo, A., Nivat, M.: Reconstruction of discrete sets from three or more X-rays. In: Bongiovanni, G., Petreschi, R., Gambosi, G. (eds.) CIAC 2000. LNCS, vol. 1767, pp. 199–210. Springer, Heidelberg (2000)
3. Barucci, D., Del Lungo, A., Nivat, M., Pinzani, R.: Reconstructing convex polyominoes from horizontal and vertical projections. *Theo. Comp. Sci.* 155, 321–347 (1996)
4. Brunetti, S., Daurat, A.: An algorithm reconstructing convex lattice sets. *Theor. Comput. Sci.* 304, 35–57 (2003)
5. Brunetti, S., Daurat, A.: Stability in Discrete Tomography: some positive results. *Disc. App. Math.* 147, 207–226 (2005)
6. Brunetti, S., Daurat, A.: Determination of Q-convex bodies by X-rays. *Elec. Notes in Dis. Math.* 20, 67–81 (2005)

7. Brunetti, S., Daurat, A.: Random generation of Q -convex sets. *Theor. Comput. Sci.* 347, 393–414 (2005)
8. Daurat, A.: Convexite dans le plan discret. Application a la tomographie. Ph.D. Thesis, LLAIC and LIAFA, Université Paris 7 (2000)
9. Daurat, A.: Determination of Q -convex sets by X-rays. *Theor. Comput. Sci.* 332, 19–45 (2005)
10. Daurat, A.: Salient Points of Q -Convex Sets. *IJPRAI* 15, 1023–1030 (2001)
11. Dulio, P., Gardner, R.J., Peri, C.: Discrete point X-rays. *J. Dis. Math.* 20, 171–188 (2006)
12. Gardner, R.J., Gritzmann, P.: Discrete tomography: determination of a finite setes by X-rays. *Trans. Am. Math. Soc.* 349(6), 2271–2295 (1997)
13. Gardner, R.J., McMullen, P.: On Hammer's X-ray problem. *J. London Math. Soc.* 21, 171–175 (1980)
14. Herman, G.T., Kuba, A. (eds.): *Discrete tomography*, pp. 1–7. Birkhäuser, Boston (1999)

Discrete Tomography Reconstruction Based on the Multi-well Potential

Tibor Lukić

Faculty of Technical Sciences, University of Novi Sad, Serbia
tiber@uns.ac.rs

Abstract. In this paper we present a new discrete tomography reconstruction algorithm developed for reconstruction of images that consist of a small number of gray levels. The proposed algorithm, called DTMWP is based on the minimization of the objective function which combines the regularized squared projection error with the multi-well potential function. The minimization is done by a gradient based method. We present experimental results obtained by application of the proposed algorithm for reconstruction of images that consist from three gray levels using small number of projections.

Keywords: Discrete tomography, multi-well potential function, image reconstruction, gradient based optimization.

1 Introduction

Tomography deals with recovering images from a number of projections. From the mathematical point of view, the object corresponds to a function and the problem posed is to reconstruct this function from its integrals or sums over subsets of its domain. In general, the tomographic reconstruction problem may be continuous or discrete. In *Discrete Tomography* (DT) [8,9] the range of the function is a finite set, in practise, DT often deals with reconstructions of images that consist of only few number of gray levels. In addition to other DT has a wide range of application in medical imaging, for example within Computer Tomography (CT), Positron Emission Tomography (PET) and Electron Tomography (ET). A special case of DT, which is called *Binary Tomography* (BT), deals with the problem of the reconstruction of a binary image.

In literature there are a variety of algebraic reconstruction methods for continuous tomography: ART, SART, SIRT, etc. For the overview of such methods we refer to [10]. Several reconstruction algorithms have been introduced for solving the DT reconstruction problem. Most of them are restricted to the BT problem only. See for example the *convex-concave regularized* reconstruction algorithm based on the DC approach [12] introduced by Schüle et al. [14], the Network Flow algorithm introduced by Batenburg [5], the Simulated Annealing based algorithm proposed by Nagy et al. [16], and the algorithms based on the Genetic and Branch and Bound optimization strategies proposed by Balázs et al. [2,3].

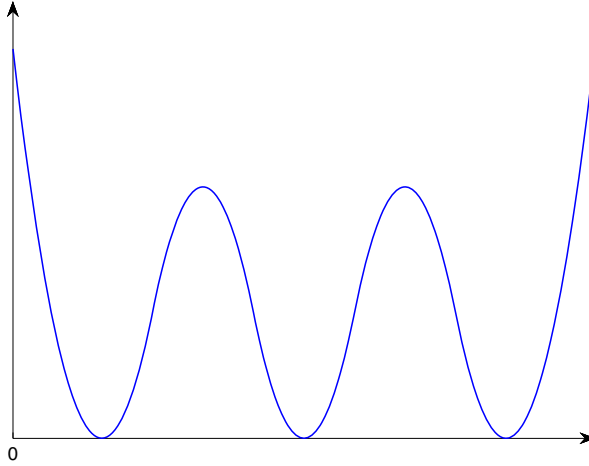


Fig. 1. Typical shape of the three-well potential function

For general DT reconstruction problem, which deals with a multi level gray image reconstructions, for best our knowledge, the heuristic *Discrete Algebraic Reconstruction Technique* (DART), introduced by Batenburg and Sijbers [6], is the only algorithm for this purpose. This algorithm combines an iterative *Algebraic Reconstruction Method* (ARM) for continuous tomography with the classical thresholding technique. The algorithm is based on the heuristic observation that after the ARM reconstruction the obtained continuous approximation of the solution approximates well enough the pixels inside the object of reconstruction, as well those one belonging to the background. The initial solution for DART algorithm is the ARM continuous solution. The DART iteration cycle consists from the following. The object boundary is determined based on the thresholding of the current solution. This thresholding is based on the given gray level values of the original image, the algorithm requires these values as an input. In the next step the ARM algorithm is performed but only on the boundary pixels, updating in this way the object boundary only. This cycle is repeated, but this time starting from the solution with previously updated object boundary, until the stop criterion is satisfied. Hence, during the reconstruction process the DART algorithm deals with the object boundary only, leaving the pixels inside and outside of the object boundary always unchanged, relying on the thresholding of the continuous ARM solution. However, thresholding is a very radical segmentation technique and its application can lead to the wrong assignments, especially in reconstructions from small number of projections when some pixel's intensities can be wrongly determined by the ARM algorithm.

We propose a new algorithm for DT reconstruction problem based on the *Multi-Well Potential* (MWP) function, see Figure 1. We minimize the objective function which is collected from the regularized squared projection error and the multi-well potential function. We use the *smooth regularization* whose application is based

on the a priori knowledge about the solution, that it is collected from compact regions of zeros and ones. This type of regularized projection error is often used in BT reconstruction algorithms, we refer to the D.C. based algorithms [14,15,17] or to the Simulated Annealing approach [16]. The reason for its “popularity” lies in the fact that its application often significantly reduces the number of needed projections. Similarly as in DART algorithm, our approach also requires as input the set of gray level values of the reconstruction. The MWP is designed to have minimums in these gray level values. Our strategy is that during the optimization process the influence of MWP term is gradually increased providing gradually stratification of the image pixel’s intensities around the given grey levels. By this approach, unlike the DART algorithm, we avoid the radical threshold technique and all pixels are affected during the whole process of the reconstruction.

The paper is organized as follows. In Section 2 we describe the DT reconstruction problem and give basic notations. In Section 3 we introduce a new method, based on the MWP function. Section 4 contains experimental results and finally, Section 5 is for conclusion remarks.

2 Reconstruction Problem

We assume a two-dimensional parallel beam projection geometry [8]. The reconstruction problem can be represented by the following linear system of equations

$$Ax = b, \quad A \in R^{m \times n}, \quad x \in \Lambda_k^n, \quad b \in R^m, \quad (1)$$

where the set $\Lambda_k = \{\mu_1, \mu_2, \dots, \mu_k\}$ and the natural number k represents a number of solution’s gray levels given by the array of values $\mu_1, \mu_2, \dots, \mu_k$. The matrix A is a so called projection matrix, whose each row corresponds to one projection ray. The corresponding components of vector b contain the detected projection values, while vector x represents the unknown image to be reconstructed. The row entries a_i of A represent the length of the intersection of pixels of the discretized volume and the corresponding projection ray, see Figure 2. Components of the vector x are discrete variables from the given level set Λ_k . In a real applications, especially for small number of projections, the system (1) is under-determined ($m < n$) and has no unique solution. Therefore, the minimization of the squared projection error

$$\min_{x \in \Lambda_k^n} \|Ax - b\|^2,$$

where by $\|\cdot\|$ we denote the Euclidean vector norm, can not lead to the satisfactory result. To avoid this problem an appropriate regularization is needed. We consider an often used smooth regularization defined by

$$\sum_i \sum_{j \in \mathcal{N}(i)} (x_i - x_j)^2, \quad (2)$$

where $\mathcal{N}(i)$ represents a set of indices of image neighbour pixels right and below from x_i . This regularization term is quadratic and convex and its role is to

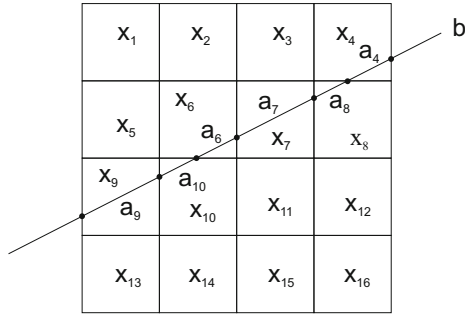


Fig. 2. The discretization model. The corresponding reconstruction problem is represented in a form of a linear system of equations, see (1).

enforce the spatial coherency of the solution. In this paper we focus on the DT problem given by

$$\min_{x \in \Lambda_k^n} \Phi_\alpha(x), \tag{3}$$

where the objective function is defined by

$$\Phi_\alpha(x) = \frac{1}{2} \left(\|Ax - b\|^2 + \alpha \sum_i \sum_{j \in \mathcal{N}(i)} (x_i - x_j)^2 \right), \tag{4}$$

parameter $\alpha > 0$ is the balancing parameter between projection error and the smoothing term. For suitable choice of α , minimization of (4) ensures both accordance of a solution with the projection data and the coherency of the solution.

From optimization point of view problem (3) represents a constrained minimization problem with discrete feasible set Λ_k which is very hard to solve. Therefore, in the next section we reformulate this problem, incorporating the MWP function, to an unconstrained problem which will be able to treat by gradient based methods.

3 Proposed Method

We transform the constrained DT problem (3) into an unconstrained optimization problem defined by

$$\min_x \left[\Phi_\alpha(x) + \gamma \cdot \sum_i W(x_i) \right], \quad \gamma > 0. \tag{5}$$

The considered MWP function W is defined by

$$W(t) = \begin{cases} (t - \mu_1)^2, & t \leq \alpha_1 \\ (t - \mu_k)^2, & t \geq \beta_{k-1} \\ (t - \mu_i)^2, & \beta_{i-1} \leq t \leq \alpha_i \\ h_i - (t - p_i)^2, & \alpha_i < t < \beta_i \end{cases}, \tag{6}$$

where for the given parameters l_i we set

$$p_i = \frac{\mu_i + \mu_{i+1}}{2}, \alpha_i = p_i - l_i, \beta_i = p_i + l_i, c_i = \frac{\mu_{i+1} - \mu_i}{2l_i} - 1, \text{ and}$$

$$h_i = \frac{1}{4}(\mu_{i+1} - \mu_i - 2l_i) \cdot (\mu_{i+1} - \mu_i) \text{ for } i = 1, \dots, k - 1.$$

Function W has k minima on the given gray level values $\mu_i \in A_k$, such that $W(\mu_i) = 0$. It is constructed to be piecewise quadratic-parabolic and smooth function, see Figure 3. Around μ_i , W is of the form $P_i(t) = (t - \mu_i)^2$ which determines a convex parabolic well. The concave parabolic junctions $Q_i(t) = h_i - (t - p_i)^2$ piecewise joins P_i and P_{i+1} , respectively at the points α_i and β_i . In order to the optimization process equally enforces neighbour gray levels, μ_i and μ_{i+1} , p_i is set to be the mean of the interval $[\mu_i, \mu_{i+1}]$ and α_i and β_i are set to be at the same distance, determined by l_i , from p_i . Values of h_i and c_i are determined in a such way that junctions at points α_i and β_i do not disturb the continuity and differentiability of W .

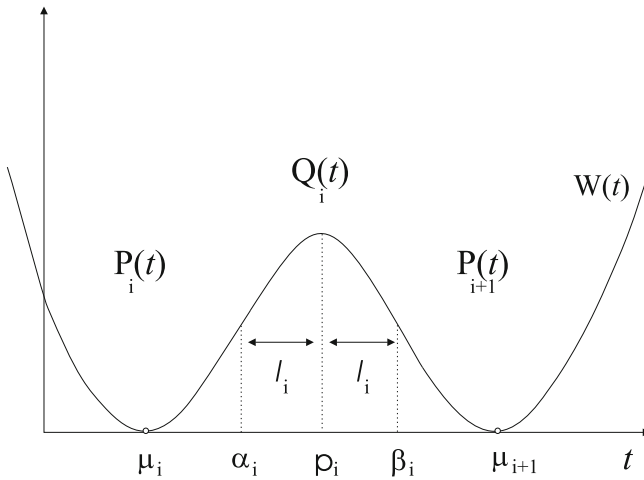


Fig. 3. Construction of the piecewise parabolic potential function

Parameters $l_i \in (0, p_i - \mu_i)$, where $i = 1, \dots, k - 1$, have to be appropriately set by the user. Two graphs of a three-well potential W for different values of l_i are presented in Figure 4. Small values of l_i make the convex parabolic wells predominant in compare with the concave parabolic junctions. However, their to small values, close to zero, can lead the optimization process, practically, to the undesirable thresholding effect. On the other hand, to large values of l_i can nullify the gray level enforcing property of the potential W . In our experimental work we set l_i as the half of the distance between μ_i and p_i .

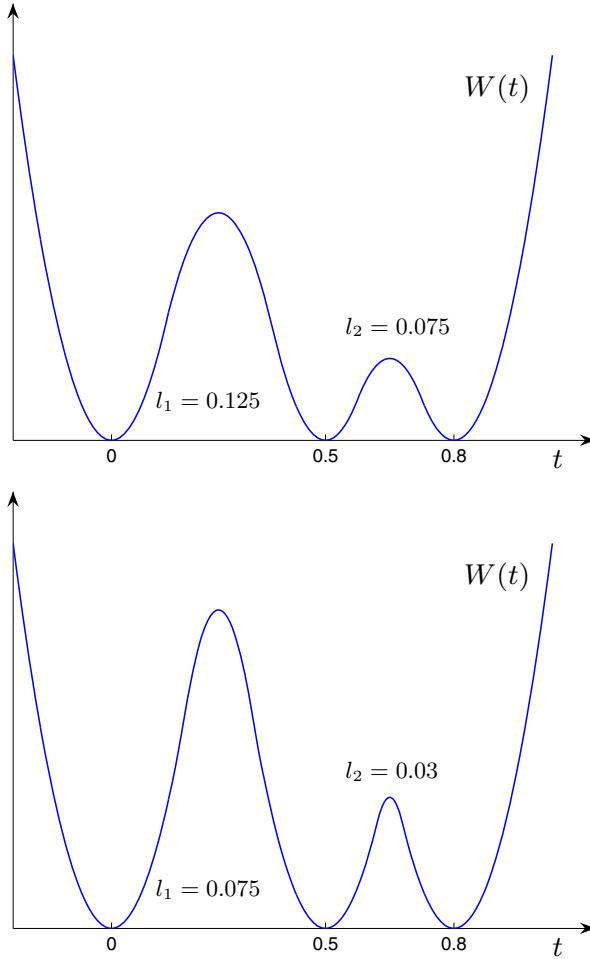


Fig. 4. Two graphs of the potential function, W for different chosen values of the parameters l_i

We note that similarly defined MWP functions such as (6) have been also proposed within other applications, we refer to the Van der Waals-Cahn-Hilliard theory of phase transitions in mechanics, see [14] or to the image classification model introduced by Samson et al. [13].

Our strategy is to solve a sequence of optimization problems (5) with gradually increasing the factor γ , which will lead to a solution consist from the gray levels according to the given set Λ_k . More precisely, we propose the *Discrete Tomography based on the MWP* (DTMWP) optimization algorithm, described below.

DTMWP Algorithm

For the given initial solution x^0 and nonnegative numbers α , γ_0 and ϵ_{out} :

$$\gamma = \gamma_0; \quad x^{init} = x^0;$$

repeat

$$x^{new} = \arg \min_x \left[\Phi_\alpha(x) + \gamma \cdot \sum_i W(x_i) \right]; \quad (7)$$

$$x^{init} = x^{new};$$

increase γ ;

until $W(x_i^{new}) < \epsilon_{out}$, $i = 1, \dots, n$.

The algorithm's initial solution x^0 is the image with all pixel values equally to a value between μ_1 and μ_k . We recommend the mean value $(\mu_1 + \mu_k)/2$, but also note that this choice does not affect significantly the algorithm's performance. Namely, in the beginning of the process, for small γ , the algorithm compute a continuous solution which after becomes an initial solution for new minimization with larger γ . In each repeat -until cycle we solve an optimization problem (7) for a fixed factor γ . The objective function of this problem is smooth and bounded below with zero, therefore there are several globally convergent line search based gradient methods which can be used for this minimization. For more details we refer to [11]. Based on our experiments we suggest the Spectral Conjugate Gradient (SCG) optimization algorithm introduced by Birgin and Martínez [7]. The solution, x^{new} which is obtained by this minimization process becomes the initial solution, x^{init} for minimization in next repeat -until cycle with increased γ . The increasing rule must be set in appropriate way: to fast increase rule can give to much significance to the stratification process and can increase the projection error; on the other hand, to slow increase rule can unnecessarily slow the convergence. The termination criterion for the repeat -until cycle, ϵ_{out} regulates the tolerance for the finally accepted (almost) gray level decomposed solution.

In the previous section we saw that Φ_α is a convex function. On the other hand, it is easy to see that W is a non-convex function. Therefore, in the beginning of the process for small γ the minimization (7) provides global minimums, but latter for larger γ the objective function in (7) becomes non-convex and we can not guaranty that DTMWP always end up in the global minimum. However, experimental results confirm its good performance.

4 Experimental Results

The proposed DTMWP algorithm is experimentally tested on the three test images (phantoms) presented in Figure 5. Reconstruction problems are composed by taking projections from different directions. We take 256 parallel rays for

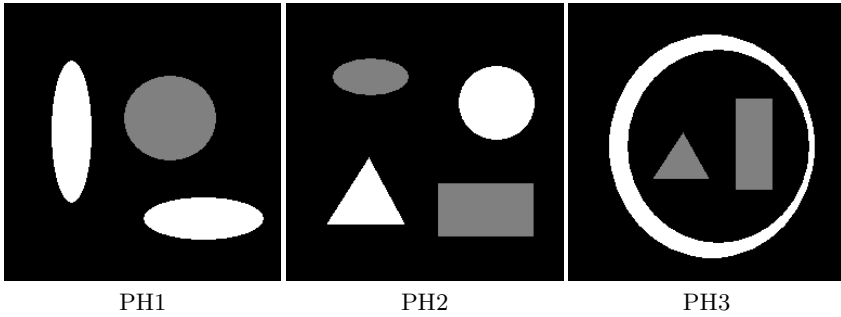


Fig. 5. Phantom images used in experiments. All images have same resolution: 256×256 . The set of gray levels, $A_3 = \{0, 0.5, 1\}$.

each projection. We distinguish reconstructions from 3, 5 and 8 projections. For 3 and 5 projections the directions are uniformly chosen within $[0^{\circ}, 90^{\circ}]$ and for 8 projections within $[0^{\circ}, 157.5^{\circ}]$.

The quality of a reconstruction is expressed by the following two error measure functions

$$E_1(x^r) = \|Ax^r - b\|,$$

$$E_2(x^r) = \sum_i |x_i^r - x_i^*|,$$

where x^r is the reconstructed image. Function E_1 measures the accordance with the projection data (projection error), while E_2 shows the distance from the original image x^* in a relation to the vector norm one.

Parameters used for the DTMWP algorithm are empirically derived and set as follows: $\gamma_0 = 0$, $\alpha = 100$ and $\epsilon_{out} = 10^{-2}$. The increasing rule for the γ factor is defined by the following quadratic type formula: $\gamma = \gamma_0 + \gamma_c \cdot i^2$, where i is an iteration counter within the repeat -until cycle and $\gamma_c = 10^{-2}$. For the potential function, W we set $l_1 = l_2 = 0.125$. The optimization problem (7) is solved by the SCG algorithm, which is fully described in [7].

The algorithm is implemented in the Matlab environment and the running time per one reconstruction is about 5 min. The obtained error measure values are reported in Table 1 and the visual look of the reconstructed images are presented in Figure 6. The method is able to provide good reconstructions of Phantoms 1 and 2 from 8 projections and reasonable ones from 5 and more

Table 1. The measured error values E_1 and E_2 of the reconstructed images

Proj.	PH1		PH2		PH3	
	$E_1(x^r)$	$E_2(x^r)$	$E_1(x^r)$	$E_2(x^r)$	$E_1(x^r)$	$E_2(x^r)$
3	119.766	6018.449	90.628	4904.557	126.095	9594.396
5	55.564	1389.369	69.731	1484.481	154.343	6163.626
8	64.173	1010.670	64.253	786.524	119.017	2675.236

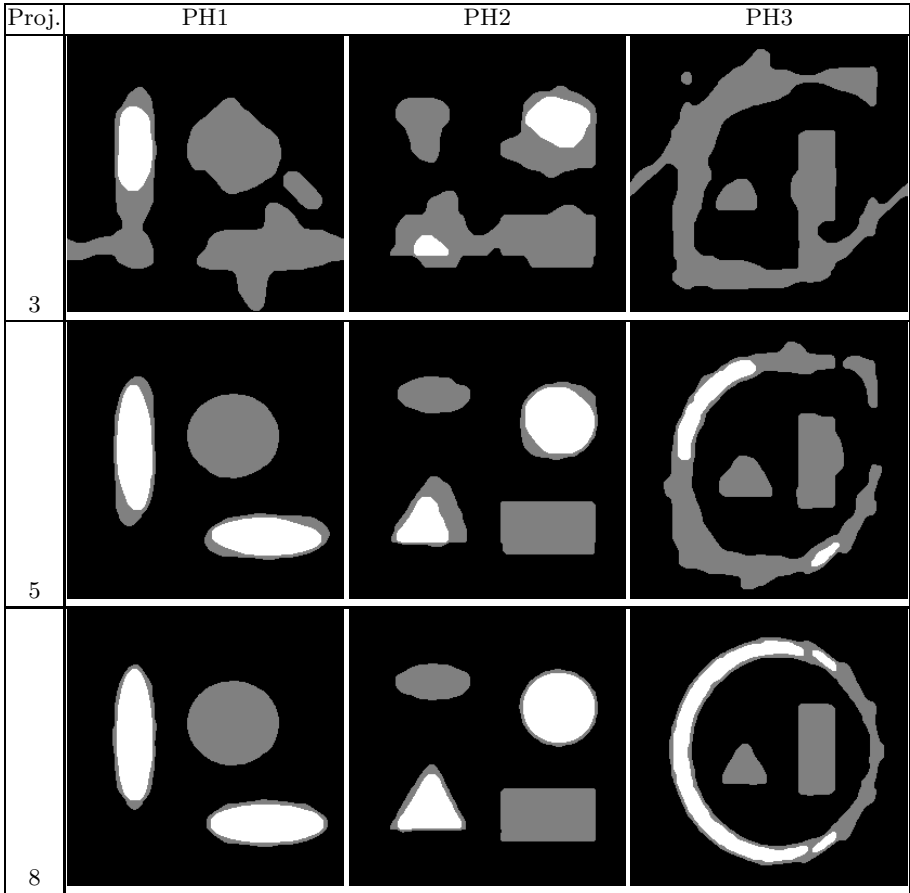


Fig. 6. Reconstructions of the phantom images presented in Figure 5. They are obtained from 3, 5 and 8 projections.

projections. The Phantom 3 is more difficult and it is obvious that for precise reconstruction more than 8 projection is needed. In general, we observe that for all test images, the quality of the reconstruction rapidly increases with number of projections.

5 Concluding Remarks

We have represented a new discrete tomography reconstruction algorithm, called DTMWP. The algorithm is based on the minimization of the objective function consists from the regularized squared projection error and the multi-well potential function. The algorithm is applicable for a reconstruction of images with multi gray levels. Our experimental results confirm the capability of the proposed

method for reconstruction of the images consist from three gray levels using a very small number of projections.

Further work can be related with inclusion an additional regularization, beside the used smooth one, with aim to accelerate the convergence and to further reduce the number of projections needed for the reconstruction process.

Acknowledgments. The author acknowledges the Ministry of Science of the Republic of Serbia for support through the Projects OI-174008 and III-44006.

References

1. Allen, S., Cahn, J.: A Microscopic Theory for Antiphase Boundary Motion and Its Application to Antiphase Domain Coarsening. *Acta Metallurgica* 27, 1085–1095 (1979)
2. Balázs, P.: Discrete Tomography Reconstruction of Binary Images with Disjoint Components Using Shape Information. *Int. Journal of Shape Modeling* 14, 189–207 (2008)
3. Balázs, P., Gara, M.: An Evolutionary Approach for Object-Based Image Reconstruction Using Learnt Priors. In: Salberg, A.-B., Hardeberg, J.Y., Jenssen, R. (eds.) SCIA 2009. LNCS, vol. 5575, pp. 520–529. Springer, Heidelberg (2009)
4. Baldo, S.: Minimal Interface Criterion for Phase Transition in Mixtures of Chan-Hillard Fluids. *Annals Inst. Henri Poincaré* 7, 67–90 (1990)
5. Batenburg, K.J.: A network flow algorithm for binary image reconstruction from few projections. In: Kuba, A., Nyúl, L.G., Palágyi, K. (eds.) DGCI 2006. LNCS, vol. 4245, pp. 86–97. Springer, Heidelberg (2006)
6. Batenburg, K.J., Sijbers, J.: DART: A Fast Heuristic Algebraic Reconstruction Algorithm for Discrete Tomography. In: *Proceedings of International Conference on Image Processing*, vol. 4, pp. 133–136 (2007)
7. Birgin, E., Martínez, J.: Spectral Conjugate Gradient Method for Unconstrained Optimization. *Appl. Math. Optimization* 43, 117–128 (2001)
8. Herman, G.T., Kuba, A.: *Discrete Tomography: Foundations, Algorithms and Applications*. Birkhäuser, Basel (1999)
9. Herman, G.T., Kuba, A.: *Advances in Discrete Tomography and Its Applications*. Birkhäuser, Basel (2006)
10. Kak, A.C., Slaney, M.: *Principles of Computerized Tomography Imaging*. SIAM, Philadelphia (2001)
11. Nocedal, J., Wright, S.J.: *Numerical Optimization*. Springer, Heidelberg (2006)
12. Pham Dinh, T., Elbeiroussi, S.: Duality in D.C (difference of convex functions) optimization, Subgradient methods. *Trends in Math. Opt.* 84, 276–294 (1988)
13. Samson, C., Blanc-Féraud, L., Aubert, G., Zerubia, J.: A Variational Model for Image Classification and Restoration. *IEEE Trans. on Pattern Analysis and Machine Intelligence* 22, 460–472 (2000)
14. Schüle, T., Schnörr, C., Weber, S., Hornegger, J.: Discrete tomography by convex-concave regularization and D.C. Programming. *Discrete Appl. Math.* 151, 229–243 (2005)
15. Schüle, T., Weber, S., Schnörr, C.: Adaptive Reconstruction of Discrete-Valued Objects from few Projections. In: *Proceedings of the Workshop on Discrete Tomography and its Applications*. *Electronic Notes in Discrete Mathematics*, vol. 20, pp. 365–384. Elsevier, Amsterdam (2005)

16. Weber, S., Nagy, A., Schüle, T., Schnörr, C., Kuba, A.: A Benchmark Evaluation of Large-Scale Optimization Approaches to Binary Tomography. In: Kuba, A., Nyúl, L.G., Palágyi, K. (eds.) DGCI 2006. LNCS, vol. 4245, pp. 146–156. Springer, Heidelberg (2006)
17. Weber, S., Schnörr, C., Schüle, T., Hornegger, J.: 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)

An Optimized Algorithm for the Evaluation of Local Singularity Exponents in Digital Signals

Oriol Pont^{1,*}, Antonio Turiel², and Hussein Yahia¹

¹ INRIA Bordeaux Sud-Ouest

351 Cours de la Libération bt. A29bis, 33405 Talence

² Institut de Ciències del Mar, ICM - CSIC

Passeig marítim de la Barceloneta 37-49, 08003 Barcelona, Catalonia

oriol.pont@inria.fr

Abstract. Recent works show that the determination of singularity exponents in images can be useful to assess their information content, and in some cases they can cast additional information about underlying physical processes. However, the concept of singularity exponent is associated to differential calculus and thus cannot be easily translated to a digital context, even using wavelets. In this work we show that a recently patented algorithm allows obtaining precise, meaningful values of singularity exponents at every point in the image by the use of a discretized combinatorial mask, which is an extension of a particular wavelet basis. This mask is defined under the hypothesis that singularity exponents are a measure not only of the degree of regularity of the image, but also of the reconstructibility of a signal from their points.

1 Introduction

Since the introduction of wavelet theory, it has been recognized that the calculation of local singularity exponents from digital signals can be used to codify them in a more compact way [10,9]. The early studies carried out over turbulent flows and other systems proved that the singularity exponents at the top points in a Wavelet Transform Modulus Maxima (WTMM) line can be easily calculated [11,12,13]. However, the extension of this methodology to any point at resolution scale was far from simple, and for those points the WTMM method become convoluted and rather imprecise [20], even for the mere assessment of the statistical properties of the signal [29].

A different approach using numerical determinations of the local gradient modulus convolved with wavelets, even with positive multiscaling bases, showed more stable results over discretized signals [25,27], leading to accurate exponents values and fine spatial resolution [21]. This new approach to singularity analysis is in the basis of the so-called Microcanonical Multiscale Formalism (MMF) [32], which has been shown to be useful to assess physical properties of turbulent flows and other multiscale systems [24,6,31,32,28,15,17,16]. It has been hence

* Corresponding author.

demonstrated that obtaining the singularity exponents at each point of a signal can reveal many useful information not only about the image, but also about the physical processes giving raise to it.

However, the performance of MMF-based singularity analysis (namely, the quality of the exponents calculated with this approach) depends on the properties of the multiscaling function used [32], the best multiscaling functions being positive functions with fast enough decay [21]. Although for statistical analysis almost any scaling function gives the same results [29], a precise geometrical determination of the underlying patterns, edges and textures requires a very fine tuning of the multiscaling function.

In this paper we discuss a method for singularity analysis introduced in a recent patent [22]. This method allows high-performance determination of the singularity exponents, with a method which is fast, computationally cheap, stable, accurate and provides fine resolution. Its definition relies in the connection of singularity exponents with the concept of image reconstruction from the Most Singular Component (MSC), as presented in [30]. The paper is structured as follows: in the next Section we introduce the basics of singularity analysis, while Section 3 explains the key concept of reconstruction from the MSC. In Section 4 the conditions to define a UPM-based measure are discussed, while Section 5 gives the settings for the calculus on reduced neighborhoods. Finally, in Section 6 our method is presented and some results shown. The last Section, Section 7, presents the conclusions of our work.

2 Definition of Singularity Analysis

Singularity analysis is a term referring to different meanings in mathematical analysis (e.g. the studies of singularities of differentiable functions); in the present work we focus on its meaning in the theory of complex systems. Using singularity analysis we intend to describe and characterize the local behavior of a \mathbb{R}^m -valued function $f(\mathbf{x})$ defined on \mathbb{R}^d around each one of its domain points \mathbf{x} according to the so-called **singularity exponent**, **Hölder exponent** [7] or **Hurst exponent** [19,8]. If the signal behaves at point \mathbf{x} according to the following limiting behavior:

$$\| f(\mathbf{x} + \mathbf{r}) - f(\mathbf{x}) \| = \alpha(\mathbf{x})r^{h(\mathbf{x})} + o(r^{h(\mathbf{x})}) \quad (\mathbf{r} \rightarrow 0) \quad (1)$$

then $h(\mathbf{x})$ is the singularity exponent at \mathbf{x} : small displacements around \mathbf{x} lead to function increment which scale as powers of the displacement modulus $r = \|\mathbf{r}\|$. A strictly n -times derivable function obviously leads to a Hölder exponent $h(\mathbf{x}) = n$, and so this formulation allows to generalize the concept of integer differentiability to real differentiability. To complete the transposition, a slightly more exigent formulation of eq. (1) is required, namely we should assume that there exists a $(1, 1)$ continuous tensor from \mathbb{R}^d to \mathbb{R}^m , $\alpha(\mathbf{x})$, such that

$$f(\mathbf{x} + \mathbf{r}) - f(\mathbf{x}) = \langle \alpha(\mathbf{x}) | \mathbf{r} \rangle r^{h(x)-1} + o(r^{h(\mathbf{x})}) \quad (\mathbf{r} \rightarrow 0) \quad (2)$$

(denoting $\langle \alpha(\mathbf{x}) | \mathbf{r} \rangle$ the standard duality bracket for $(1, 1)$ tensors). When such representation is possible, the exponent $h(\mathbf{x})$ is called the Hurst exponent of the function; if only eq. (II) can be applied we will prefer to speak about Hölder exponents.

Assessment based on Hölder exponents can only be applied to very specific signals; in general, the presence of long range correlations and the effects of noise and discretization would preclude a direct evaluation of the scaling exponent [27,32].

A more general framework is given by singularity exponents, which are defined using gradient-based measures [27]. Given a signal $s(\mathbf{x})$, defined in \mathbb{R}^d and with values in \mathbb{R} , we can define its associated gradient measure μ by its density $d\mu(\mathbf{x})$, which is given by:

$$d\mu(\mathbf{x}) = \|\nabla s\|(\mathbf{x}) \, d\mathbf{x} \tag{3}$$

This measure is by definition absolutely continuous with respect to Lebesgue measure. Hence, the measure of any Borelian \mathcal{A} is given by:

$$\mu(\mathcal{A}) = \int_{\mathcal{A}} d\mathbf{x} \|\nabla s\|(\mathbf{x}) \tag{4}$$

Gradient measures also allow to characterize the local singularity of any point, and in a direction-independent manner. Following eq. (II), let us consider a function $f(\mathbf{x})$ with a Hölder exponent $h(\mathbf{x}) + 1$ at a point \mathbf{x} (notice the shift $+1$ introduced for later convenience). Let $\mathcal{B}_r(\mathbf{x})$ be the ball (using an arbitrary norm in \mathbb{R}^d) of radius r centered around \mathbf{x} . So, we obtain [27]:

$$\mu(\mathcal{B}_r(\mathbf{x})) = \alpha(\mathbf{x}) r^{d+h(\mathbf{x})} + o\left(r^{d+h(\mathbf{x})}\right) \quad (r \rightarrow 0) \tag{5}$$

where d is the dimension of the domain space ($d = 2$ in images). The introduction of gradient measures is convenient, as measures can also be wavelet-projected to obtain smooth interpolations from discretized data. Given a wavelet Ψ , we define [3,12] the wavelet projection of the measure μ at the point \mathbf{x} and scale r , denoted by $\mathcal{T}_\Psi \mu(\mathbf{x}, r)$, as:

$$\mathcal{T}_\Psi \mu(\mathbf{x}, r) = \int_{\mathbb{R}^d} d\mu(\mathbf{x}') \frac{1}{r^d} \Psi\left(\frac{\mathbf{x} - \mathbf{x}'}{r}\right) \tag{6}$$

i.e. operator \mathcal{T}_Ψ is a map from the set \mathcal{M} of σ -finite measures on \mathbb{R}^d to the set of functions $\mathbb{R}^d \times \mathbb{R}^+ \rightarrow \mathbb{R}$. If the signal possesses a singularity exponent at the point \mathbf{x} according to eq. (5), then the wavelet projections allow to infer this same exponent, as they verify [3,27]:

$$\mathcal{T}_\Psi \mu(\mathbf{x}, r) = \alpha_\Psi(\mathbf{x}) r^{h(\mathbf{x})} + o\left(r^{h(\mathbf{x})}\right) \quad (r \rightarrow 0) \tag{7}$$

The main advantage of using measures over discretized data is that any function Ψ can be used to evaluate singularity exponents using eq. (7), even positive functions [32]. As discussed in [29], the resolution capability of a wavelet depends on the number of zero-crossings it has, which is increased in higher-order

wavelets but is minimum for positive wavelets. So, gradient measures improve the spatial resolution of singularity exponents [21].

3 MSC and Its Connection with Reconstruction

An important ingredient in the construction of wavelets with optimized resolution capability is the concept of reconstruction of signals from partial information about its gradient. The theoretical and practical implementation of this reconstruction algorithm was first introduced in [30] (see discussion there).

We consider signals having a singularity exponent $h(\mathbf{x})$ at each point [14,5] and for which these exponents are organized forming sets with a particular multiscale structure [4]. That is, the values of the singularity exponents do not take arbitrary values but must be organized so that they define a hierarchy of multiscale geometrical structures “matching” and realizing closely the cascading properties of some random variables associated to the macroscopic description of the system under study [18]. Due to the difficulties of classical methodologies to assign a precise value of singularity exponent $h(\mathbf{x})$ to each point, all the characterizations of this hierarchy that have been tried up to now are merely statistical. In [30] a new question was posed: if the hierarchy truly exists in complex signals, can they be reconstructed starting from the vertex of this hierarchy? For multifractals, the set associated to the vertex is well-known, at least from the theoretical point of view: it is the so-called **Most Singular Component** (MSC), which is the set comprising the points with most singular (*i.e.*, most negative) values of $h(\mathbf{x})$ [27,32].

The thesis in [30] is that the MSC contains enough information to fully reconstruct the signal (in that reference, the reconstruction of images is analyzed, although the formulas are valid for any number of dimensions). As we are working with gradient measures, the data to be retained at the MSC is the gradient of the signal. So, it was hypothesized that there exists an universal operator to reconstruct signals starting from the values of the signal, and leading to a reconstruction algorithm consistent with the known statistical invariances of turbulence and multiscale signals [5]. The algorithm was required to be deterministic, linear, translational invariant, isotropic and leading to the known power-spectrum shape. Under these requirements, it turned out that there exists, if any, only one possible operator to reconstruct signals from the gradient on the MSC. Let us first define a convenient notation for the starting data. For a given multiscale signal s let us denote by \mathcal{F}_∞ the MSC, that is to say \mathcal{F}_∞ is the set of points \mathbf{x} such that $h(\mathbf{x}) \in]h_\infty - \Delta, h_\infty + \Delta[$ with h_∞ being the minimum value of all $h(\mathbf{x})$ over the finite domain of the discrete signal, and Δ a threshold parameter; we will define the essential gradient of s , $\nabla_{\mathcal{F}_\infty} s$, as follows:

$$\nabla_{\mathcal{F}_\infty} s(\mathbf{x}) = \nabla s(\mathbf{x}) \delta_{\mathcal{F}_\infty}(\mathbf{x}) \tag{8}$$

where $\delta_{\mathcal{F}_\infty}$ is a delta distribution associated to the continuum of the \mathcal{F}_∞ , homogeneous in (Hausdorff) topological dimension to a repartition in between

dimensions $d - 1$ and d : it assigns uniform weight to the points on the MSC \mathcal{F}_∞ and vanishes outside the MSC. According to this notation, the reconstruction formula [30] reads:

$$s(\mathbf{x}) = (\mathbf{g} \cdot \nabla_{\mathcal{F}_\infty} s)(\mathbf{x}) \quad (9)$$

where the symbol \cdot means convolution dot-product of vectors and the vector field \mathbf{g} is the universal reconstruction kernel, which can be easily expressed in Fourier space, namely:

$$\hat{\mathbf{g}}(\mathbf{k}) = i \frac{\mathbf{k}}{\|\mathbf{k}\|^2} \quad (10)$$

and $i = \sqrt{-1}$ is the imaginary unit. So defined, the reconstruction kernel \mathbf{g} is a kind of inverse gradient operator. There is always a set \mathcal{F}_∞ from which reconstruction is perfect, the whole domain: If $\mathcal{F}_\infty = \mathbb{R}^d$, then eq. (9) reduces to the trivial identity $\nabla_{\mathcal{F}_\infty} s = \nabla s$. But from eq. (9) is not evident if there exists a smaller set $\mathcal{F}_\infty \subset \mathbb{R}^d$ such that reconstruction is also perfect. Following the derivation in [30], we can conclude that any set \mathcal{F} leading to a perfect reconstruction must verify:

$$\operatorname{div} \left(\nabla_{\mathcal{F}^c} s \right) = 0 \quad (11)$$

where \mathcal{F}^c is the complementary set of \mathcal{F} . As the divergence operator is local and the formula above is linear, the decision to include or not a point can be taken on the basis of any neighborhood around that point. The points that must always be included to obtain a perfect reconstruction are hence those with values that cannot be predicted just knowing the values in their surroundings; they are hence called unpredictable points (in opposition to the other points, which are predictable). Predictability is a subject at the core of the analysis of complex systems and signals [21], where for instance Lyapunov exponents and Kolmogorov-Sinai entropy are known measures of information growth in a dynamical system. The set \mathcal{F}_u formed by the collection of all the unpredictable points is what we will call the Unpredictable Points Manifold (UPM) and it is, by definition, the smallest set for which eq. (9) lead to a perfect reconstruction. The hypothesis in [30] is that $\mathcal{F}_u = \mathcal{F}_\infty$, a conjecture which is at the base of the framework of *reconstructible systems*. What is evident from experiences is that the MSC leads to good reconstructions (see discussion in [32]).

4 General Conditions to Define UPM-Measures

We now step forward to generalize the concept of gradient measure introduced in the previous sections to the novel concept of UPM-measure. The basic requirements to define a singular positive UPM-measure μ are:

- i) It is concerned with the local singular behavior of functions.
- ii) It leads to a MSC as close to the UPM as possible.

In some sense, UPM-measures are gradient measures which also take into account the degree of predictability of points according to eq. (11). So that, they take the form of a gradient (in the sense of finite difference over discretized signals) but with penalty terms associated to the lack of predictability. The best way to keep on working around singularities is to define UPM-measures as vectorial wavelet projections of standard gradient measures. So, the UPM-measure is a carefully designed vectorial wavelet projection of the gradient measure so that it penalizes unpredictability.

In our method, in contrast with standard singularity analysis, we will not perform many wavelet projections of the UPM measure in order to extract the singularity exponents by means of a log-log regression applied to eq. (7). Wavelet-projecting the measure at several scales is costly in computer time and only serves to enhance the resolution of less singular structures at the cost of coarsening most singular ones (see a discussion on this in [29]). But as we are mainly interested in the most singular structures, it is hence harmful to our interests to project across multiple scales. Instead, we will make use of point estimates [29,15] of the singularity exponents, namely:

$$h(\mathbf{x}) = \frac{\log(\mathcal{T}_\Psi \mu(\mathbf{x}, r_0) / \langle \mathcal{T}_\Psi \mu(\cdot, r_0) \rangle)}{\log r_0} + o\left(\frac{1}{\log r_0}\right) \tag{12}$$

where $\langle \mathcal{T}_\Psi \mu(\cdot, r_0) \rangle$ is the average value of the wavelet projection over the whole signal and serves to diminish the relative amplitude of the $o\left(\frac{1}{\log r_0}\right)$ correction. When applying eq. (12) we will need that r_0 is small enough to neglect this correction. The scale r_0 will be defined as the smallest accessible one, that is, the pixel scale. We conventionally assign a Lebesgue measure of 1 to the whole space domain, so for a $N \times M$ image the value of r_0 is fixed to $r_0 = \frac{1}{\sqrt{NM}}$, so in general we need that images are large enough to make the first term in the right hand side of eq. (12) a good approximation of the singularity exponent. In practical terms, this implies a resolution around 100×100 pixels or larger.

5 Calculus on Reduced Neighborhoods: Cross Fourier Transform

In order to assess the degree of predictability of a given point, we will apply the reconstruction formula, eq. (9), for the smallest possible neighbor of a point, namely its $2d$ nearest neighbors in $2d$ connexity neighborhoods. In 2D ($d = 2$) this consists of 4 neighbors, that with the point altogether form a cross. For any quantity $p(\mathbf{x})$ we will represent the neighborhood of any point \mathbf{x}_0 by a 5-component vector comprising this point and its 4 nearest neighbors, following the indexing convention established in Figure 1. So, the central point will be assigned the index 0, the point at its right will be indexed 1, the one on the left is indexed 2, that on top is indexed as 3 and the one on the bottom is indexed 4. So, we convert the neighbor in the vector $(p_0, p_1, p_2, p_3, p_4)$. The notion of predictable point easily extends to any number of dimensions, regardless of the

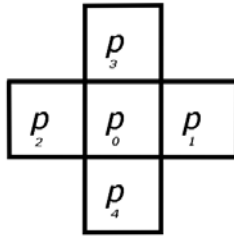


Fig. 1. Schematic representation of the indexing of the points in the 2D cross

number of components of the neighbor vector, which grows as the dimension d increases.

We could apply the harmonics of the standard Fourier Transform on the discrete signal $(p_0, p_1, p_2, p_3, p_4)$, but this is not a good idea. The harmonics of the standard Fourier Transform (i.e., $(e^{2ik\pi/n})_k$) depend on the size of the embedding space, so they would lead to a dimension-dependent measure of the predictability. To overcome this problem, we note that relative to the center of the cross, the position of the other points correspond to displacements of ± 1 (in pixel units) either in the x -direction or in the y -direction. So that, to define a special type of Fourier transform specialized to this cross formation, the basic Nyquist frequency in each direction is $2\pi/3$. Consequently we introduce

$$j = e^{2\pi i/3} = \cos(2\pi/3) + i \sin(2\pi/3) = -\frac{1}{2} + i\frac{\sqrt{3}}{2}, \bar{j} = j^2 \quad (13)$$

We define the direct Cross Fourier Transform of any 5-vector $\mathbf{p} = (p_0, p_1, p_2, p_3, p_4)$ as the complex 5-vector $\hat{\mathbf{p}} = (\hat{p}_0, \hat{p}_1, \hat{p}_2, \hat{p}_3, \hat{p}_4)$ obtained according to the following formula:

$$\hat{\mathbf{p}} = \mathbb{F} \mathbf{p} \quad (14)$$

where \mathbb{F} is the following 5×5 complex matrix:

$$\mathbb{F} = \frac{1}{3} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & \bar{j} & \bar{j} & 1 & 1 \\ 1 & \bar{j} & j & 1 & 1 \\ 1 & 1 & 1 & j & \bar{j} \\ 1 & 1 & 1 & \bar{j} & j \end{bmatrix} \quad (15)$$

This matrix represents the linear combination of the harmonics associated to the displacements in the cross and is designed to represent with the maximum fidelity the composition at the center of the cross, starting from the nearest points. The inverse of this matrix is:

$$\mathbb{F}^{-1} = \begin{bmatrix} -1 & 1 & 1 & 1 & 1 \\ 1 & \bar{j} & j & 0 & 0 \\ 1 & j & \bar{j} & 0 & 0 \\ 1 & 0 & 0 & \bar{j} & j \\ 1 & 0 & 0 & j & \bar{j} \end{bmatrix} \tag{16}$$

We need to define surrogates of the gradient and the reconstruction formula restricted to the cross neighborhood, in order to evaluate in a fast way the degree of predictability of the central point. For that reason, we will construct appropriate implementations of the gradient and of the gradient reconstruction formula, based on the Cross Fourier Transform.

The **Cross Gradient Operator** is the operator $(\partial_x, \partial_y) = \mathbb{F}^{-1} \cdot (\hat{\partial}_x, \hat{\partial}_y) \cdot \mathbb{F}$. In Fourier space the operator acts by simply multiplying any function by the functions $\hat{\partial}_x$ and $\hat{\partial}_y$ to obtain the x and the y coordinate, respectively. The function $\hat{\partial}_x$ is defined as:

$$\hat{\partial}_x = (0, i\sqrt{3}, -i\sqrt{3}, 0, 0) \tag{17}$$

and analogously we have:

$$\hat{\partial}_y = (0, 0, 0, i\sqrt{3}, -i\sqrt{3}) \tag{18}$$

The **Cross Reconstruction Operator** is one of the inverses of the Cross Gradient Operator. As the gradient operator eliminates any constant summed up to each component of the 5-vector representing the neighborhood, the reconstruction is defined up to a constant shift; our implementation of the cross reconstruction operator is such that the 5-vector has zero mean, $\sum_{i=1}^5 s_i = 0$. For that reason, signals should have the mean subtracted before applying these two operators (see below).

The Cross Reconstruction is the operator $R = \mathbb{F}^{-1} \cdot \hat{R} \cdot \mathbb{F}$. In Fourier space \hat{R} has two functional components, $\hat{R} = (\hat{R}_x, \hat{R}_y)$; the operator acts as the sum of the product of each component with the corresponding component (x and y) of the gradient on which it is operated. The component \hat{R}_x is defined as:

$$\hat{R}_x = (0, -i/\sqrt{3}, i/\sqrt{3}, 0, 0) \tag{19}$$

and analogously for \hat{R}_y ,

$$\hat{R}_y = (0, 0, 0, -i/\sqrt{3}, i/\sqrt{3}) \tag{20}$$

The Cross Gradient and the Cross Reconstruction are the two basic algorithms for the design of the UPM-measures. They can be simplified to a 5×5 matricial form, for faster numeric implementation.

6 Local Correlation Singularity Measure

The **Local Correlation Singularity Measure** is designed to measure the unpredictability of a given point, just quantifying the difference on the actual

value of the detrended (*i.e.*, after subtracting the mean) signal at a given point and the inferred one from their four neighbors. It is defined algorithmically as follows:

Goal: To evaluate $\mathcal{T}_{\Psi_{l_{csm}}}\mu(\mathbf{x}_0, r_0)$ at a given point \mathbf{x}_0 .

Algorithm

1. The neighborhood of \mathbf{x}_0 is converted into a 5-vector $\mathbf{s} = (s_0, s_1, s_2, s_3, s_4)$ according to the scheme in Figure 1.
2. The vector is conveniently detrended: we first obtain $\bar{S} = \frac{1}{3} \sum_{i=1}^5 s_i$, and we define the detrended vector, $\mathbf{p} = (p_0, p_1, p_2, p_3, p_4)$ as:

$$p_0 = s_0 + \bar{S} ; p_i = s_i - \bar{S} , i = 1, \dots, 4$$

3. We apply the Cross Gradient Operator to \mathbf{p} , so we obtain the vector \mathbf{g}_x and \mathbf{g}_y .



Fig. 2. **Top:** Left: Original Lena image; Right: Singularity exponents estimated using the Local Correlation Singularity Measure; they are represented using a inverse grayscale palette (the brightest the smaller, so more singular). **Bottom:** Left: MSC, defined as $\{h < -0.5\}$; it comprises 30% of the points of the image ; Right: Reconstruction. Some details are missing due to the lack of capability of the method to capture every UPM point; however, the reconstruction is of high quality (24.5 dB).

4. We keep the value of the first components of these two vectors for a later use, $A_x = g_{x,0}$, $A_y = g_{y,0}$.
5. We set these two components to zero, $g_{x,0} = g_{y,0} = 0$.
6. We apply the Cross Reconstruction Operator to the resulting vectors \mathbf{g}_x and \mathbf{g}_y , to obtain the reconstructed signal \mathbf{r} .
7. We apply once more the Cross Gradient Operator onto \mathbf{r} to obtain $\boldsymbol{\rho}_x$ and $\boldsymbol{\rho}_y$.
8. We define the Local Correlation Singularity Measure as the modulus of the difference of the cross gradients at the center of the cross, namely:

$$\mathcal{T}_{\psi_{lcsm}}\mu(\mathbf{x}_0, r_0) = \sqrt{(A_x - \rho_{x,0})^2 + (A_y - \rho_{y,0})^2}$$

In fact, this last step means to keep the modulus of a vector-valued wavelet projection, but to simplify notation we leave it as is.

9. The singularity exponent $h(\mathbf{x}_0)$ is then obtained in application of eq. (12).

In Figure 2 we show an example of the application of the singularity analysis based on the Local Correlation Singularity Measure.

7 Conclusions

The accurate estimation of singularity exponents in multiscale systems allows characterizing their relevant features and identifying their information content. This is particularly important for the case of digital images, where the degree of singularity is directly related to the distribution of information, and so its knowledge can be used for compact coding or reconstructing from the Most Singular Component (MSC). However, digital images are discretized and this fact is an important obstacle for precisely retrieving its singularity exponents: even when using wavelet projections, most standard wavelet bases only give average results.

In this article, we have presented a recent algorithm that allows obtaining the singularity exponents of an image at every point. The singularity exponents are extracted in a precise and meaningful way, by means of a discretized combinatorial mask. This mask is constructed by considering the singularity exponent of a given point as both a measure of the singularity/regularity degree and a measure of the unpredictability of that point. The result is a discretized, numerical extension of a particular wavelet basis.

We have presented and discussed the method for singularity analysis noted as ‘‘Local Correlation Singularity Measure’’ in patent [22]. This method attains at the same time good quality and spatial resolution in the estimation of singularity exponents. Additionally, the reconstruction from the MSC is of high quality. As an illustration, we have shown the singularity exponents from the Local Correlation Singularity Measure and the reconstruction from their MSC for Lena’s image, for which the reconstruction quality is of 24.5 dB. The prospects of this method includes image compression [30], assessment of streamlines in turbulent

flows [31,26], detection of convection meteorological systems [23] or detection of investment cycles in stock market series [28], among others.

The presented methodology for 2D images can be easily generalized to any dimensionality. In addition, it is possible to define other UPM-measures other than the Local Correlation Singularity Measure, that can give better performance in certain cases. All these additional developments and their respective applications will be the object of future communications.

References

1. Aurell, E., Boffetta, G., Crisanti, A., Paladin, G., Vulpiani, A.: Predictability in the large: an extension of the concept of lyapunov exponent. *Journal of Physics A* 30, 1–26 (1997)
2. Boffetta, G., Cencini, M., Falcioni, M., Vulpiani, A.: Predictability: a way to characterize complexity. *Physics reports* 356(6), 367–474 (2001)
3. Daubechies, I.: Ten lectures on wavelets. CBMS-NSF Series in App. Math. Capital City Press, Montpellier (1992)
4. Falconer, K.: *Fractal Geometry: Mathematical Foundations and Applications*. John Wiley and Sons, Chichester (1990)
5. Frisch, U.: *Turbulence: The legacy of A.N. Kolmogorov*. Cambridge Univ. Press, Cambridge (1995)
6. Isern-Fontanet, J., Turiel, A., Garcia-Ladona, E., Font, J.: Microcanonical multifractal formalism: application to the estimation of ocean surface velocities. *Journal of Geophysical Research* 112, C05024 (2007)
7. Jaffard, S.: Multifractal formalism for functions. I. Results valid for all functions. *SIAM Journal of Mathematical Analysis* 28(4), 944–970 (1997)
8. Jones, C.L., Lonergan, G.T., Mainwaring, D.E.: Wavelet packet computation of the hurst exponent. *J. Phys. A: Math. Gen.* 29(10), 2509 (1996)
9. Mallat, S., Huang, W.L.: Singularity detection and processing with wavelets. *IEEE Trans. in Inf. Th.* 38, 617–643 (1992)
10. Mallat, S., Zhong, S.: Wavelet transform maxima and multiscale edges. In: Ruskai, M.B., et al. (eds.) *Wavelets and their Applications*. Jones and Bartlett, Boston (1991)
11. Mallat, S., Zhong, S.: Characterization of signals from multiscale edges. *IEEE Trans. on Pattern Analysis and Machine Intelligence* 14, 710–732 (1992)
12. Mallat, S.: *A Wavelet Tour of Signal Processing*, 2nd edn. Academic Press, London (1999)
13. Muzy, J.F., Bacry, E., Arneodo, A.: Wavelets and multifractal formalism for singular signals: Application to turbulence data. *Physical Review Letters* 67, 3515–3518 (1991)
14. Parisi, G., Frisch, U.: On the singularity structure of fully developed turbulence. In: Ghil, M., Benzi, R., Parisi, G. (eds.) *Turbulence and Predictability in Geophysical Fluid Dynamics*. Proc. Intl. School of Physics E. Fermi, pp. 84–87. North Holland, Amsterdam (1985)
15. Pont, O., Turiel, A., Pérez-Vicente, C.: Application of the microcanonical multifractal formalism to monofractal systems. *Physical Review E* 74, 61110 (2006)
16. Pont, O., Turiel, A., Pérez-Vicente, C.: Description, modeling and forecasting of data with optimal wavelets. *Journal of Economic Interaction and Coordination* 4, 39–54 (2009)

17. Pont, O., Turiel, A., Pérez-Vicente, C.: Empirical evidences of a common multifractal signature in economic, biological and physical systems. *Physica A* 388, 2025–2035 (2009)
18. She, Z.S., Leveque, E.: Universal scaling laws in fully developed turbulence. *Physical Review Letters* 72, 336–339 (1994)
19. Simonsen, I., Hansen, A., Magnar, O.: Determination of the hurst exponent by use of wavelet transforms. *Phys. Rev. E* 58(3), 2779–2787 (1998)
20. Struzik, Z.R.: Determining local singularity strengths and their spectra with the wavelet transform. *Fractals* 8(2), 163–179 (2000)
21. Turiel, A.: Relevance of multifractal textures in static images. *Electronic Letters on Computer Vision and Image Analysis* 1(1), 35–49 (2003)
22. Turiel, A.: Method and system for the singularity analysis of digital signals, patent registered under number PCT/ES2008/070195 (2008)
23. Turiel, A., Grazzini, J., Yahia, H.: Multiscale techniques for the detection of precipitation using thermal IR satellite images. *IEEE Geoscience and Remote Sensing Letters* 2(4), 447–450 (2005), doi:10.1109/LGRS.2005.852712
24. Turiel, A., Isern-Fontanet, J., García-Ladona, E., Font, J.: Multifractal method for the instantaneous evaluation of the stream function in geophysical flows. *Physical Review Letters* 95(10), 104502 (2005), doi:10.1103/PhysRevLett.95.104502
25. Turiel, A., Mato, G., Parga, N., Nadal, J.P.: The self-similarity properties of natural images resemble those of turbulent flows. *Physical Review Letters* 80, 1098–1101 (1998)
26. Turiel, A., Nieves, V., García-Ladona, E., Font, J., Rio, M.H., Larnicol, G.: The multifractal structure of satellite temperature images can be used to obtain global maps of ocean currents. *Ocean Science* 5, 447–460 (2009)
27. Turiel, A., Parga, N.: The multi-fractal structure of contrast changes in natural images: from sharp edges to textures. *Neural Computation* 12, 763–793 (2000)
28. Turiel, A., Pérez-Vicente, C.: Role of multifractal sources in the analysis of stock market time series. *Physica A* 355, 475–496 (2005)
29. Turiel, A., Pérez-Vicente, C., Grazzini, J.: Numerical methods for the estimation of multifractal singularity spectra on sampled data: a comparative study. *Journal of Computational Physics* 216(1), 362–390 (2006)
30. Turiel, A., del Pozo, A.: Reconstructing images from their most singular fractal manifold. *IEEE Trans. on Im. Proc.* 11, 345–350 (2002)
31. Turiel, A., Solé, J., Nieves, V., Ballabrera-Poy, J., García-Ladona, E.: Tracking oceanic currents by singularity analysis of micro-wave sea surface temperature images. *Remote Sensing of Environment* 112, 2246–2260 (2008)
32. Turiel, A., Yahia, H., Pérez-Vicente, C.: Microcanonical multifractal formalism: a geometrical approach to multifractal systems. Part I: Singularity analysis. *Journal of Physics A* 41, 15501 (2008)

Community Detection for Hierarchical Image Segmentation

Arnaud Browet, P.-A. Absil, and Paul Van Dooren

ICTEAM Institute, Université catholique de Louvain, Louvain-la-Neuve, Belgium
{arnaud.browet,paul.vandooren}@uclouvain.be,
absil@inma.ucl.ac.be

Abstract. In this paper, we present a new graph-based technique to detect segments or contours of objects in a given picture. Our algorithm is designed as an approximation of the Louvain method that unfolds the community structures in a large graph. Without any a priori knowledge on the input picture, relevant regions are extracted while the optimal definition of a contour, depending on the user or the application, can be tuned using parameters. The communities found are also hierarchical allowing to find subregions inside an object. We present experimental results of our method on real images.

Keywords: Image segmentation, community detection, modularity optimization.

1 Introduction

How many objects are present in a picture? This simple question from a perception point of view (depending on what one considers to be an object) has raised a lot of research for the last decades and remains a very challenging problem for a computer. There are many applications where this consideration takes place: discovering abnormal shadows on a CT or PET scan for tumor detection, detection of people and objects from images of surveillance cameras or from a camera at the front of a car in the context of collision detection, etc.

The answer is never unique in image segmentation because defining the objects of interest is user or application dependent. However, we want to address the question of image segmentation without any a priori knowledge about the shape, the position or the number of objects displayed in the input picture. Nevertheless, we will keep a number of tuning parameters to optimize the method depending on some properties of the picture, like the number of pixels or their range. Based on a graph built from the image, we will show that finding the relevant structure of objects is possible using the optimization of a scalar function called the modularity. This measure was introduced by Newman and Girvan [16] and represents the quality of clusters defined on the nodes of the graph. Unfortunately, like the definition of clusters, this measure is combinatorial and the optimization of this criterion cannot be done in polynomial time. We will present a greedy algorithm that provides a good approximation of the optimal

modularity. This algorithm will allow us to define communities of nodes in the graph which lead to coherent groups or subgroups of pixels in the image.

The paper is organized as follows: in section 2 we review some classical graph-based techniques for image segmentation and introduce our notations. Then in section 3 we review the concept of modularity for community detection in large graphs and describe our algorithm to approximate the optimal modularity. In section 4 we show some experimental results on real images. Finally, section 5 concludes with the main results of this paper and proposes future work.

2 Related Work

Since the number of techniques in computer vision is quite large, each one coming from various fields (histograms metric [21], watershed techniques [3], active contours [13,14], manifold learning [12,19], etc), we will restrict our review to graph-based techniques. Those procedures generally build a graph G where each node represents a pixel in the input picture. Since the number of pixels is very large, even for low resolution pictures, ideas have been proposed to reduce the number of nodes in the graph to a representative subsample of pixels or regions, for example in [2] or see other references therein.

Each weighted edge of the graph G represents the similarity between a pair of pixels and is stored in the weighted adjacency matrix W . Without a priori knowledge about the components of the input image, a classical way to define an edge weight is

$$w_{ij} = \begin{cases} e^{-\frac{d(i,j)^2}{\sigma_x^2}} e^{-\frac{|F(i)-F(j)|^2}{\sigma_i^2}} & \text{if } d(i,j) < d_{max}, \\ 0 & \text{otherwise,} \end{cases} \quad (1)$$

where $d(i,j)$ is the distance between pixels i and j (e.g. the Euclidean or the Chebyshev distance) and $F(i)$ is a feature vector evaluated at pixel i . This feature vector can be for example the scalar intensity value for gray scaled images or the HSV transform for color images. The edge weight is controlled by the user defined parameters σ_x , σ_i and d_{max} .

Based on this undirected weighted graph, classical methods [5,18,20,22] try to find a selection of edges, called a cut, to optimize a criterion. A cut is defined as a set of edges that creates disconnected components when removed from the graph. The cost function is in general the total weight of the removed edges with an additional scaling that penalizes the creation of very small components: in the ratio cut the scale is the dimension of the components (Cox *et al.* [5]), in the minimum mean cut the scale is the number of cut edges (Wang & Siskind [20]) and in the normalized cut the scale is the internal similarity of the components (Shi & Malik [18]). Another efficient method has been proposed by Felzenszwalb and Huttenlocher [7] that extracts regions in the graph minimizing what they call internal difference of regions while maximizing the external difference between regions. Unfortunately, in general these methods need to know the number of objects of interest in the input picture or they have to set an arbitrary threshold

on the minimum value of the cut criterion and this may lead to an inappropriate segmentation. In particular, optimizing the normalized cut criterion is known to be NP-hard [18] but a continuous relaxation can be solved in polynomial time using spectral graph theory. Some recent works [19] propose to reduce the computational cost of such methods by computing an approximation of the optimal cut criterion using the incomplete Cholesky decomposition.

Those methods try to split the large pixel graph into salient regions by defining boundaries between the regions. On the other hand, one can consider the opposite way of grouping adjacent connected nodes, defining the salient regions and letting the boundaries appear by themselves. We propose to analyze this idea by considering the previously constructed graph as a large (social) network divided in communities. Communities as introduced by Newman and Girvan [10] are defined as sets of highly connected nodes with only a few or light connections between distinct groups. This informal definition is obviously not sufficient to identify the community structure in a graph. Therefore Newman and Girvan [16] introduce a scalar function Q called the modularity to evaluate the quality of a community structure in a graph:

$$Q = \frac{1}{2m} \sum_{i,j=1}^n (w_{ij} - N_{ij}) \delta(C_i, C_j) , \quad (2)$$

where n is the number of nodes in the graph, w_{ij} is the edge weight as previously defined in (1), N_{ij} is a null model between nodes i and j , C_i is the community of node i and $\delta(C_i, C_j) = 1$ if nodes i and j are in the same community and 0 otherwise. Here m is the sum of the weights of all the edges in the graph and is a scaling parameter ensuring that $Q \in [-1, 1]$. The null model N_{ij} is designed to assess the strength of an edge and can be defined as the expected weight of the edge between nodes i and j , given that the degree of each node is known:

$$N_{ij} = \frac{k_i k_j}{2m} . \quad (3)$$

Here k_i is the weighted degree of node i i.e. $k_i = \sum_{j=1}^n w_{ij}$. The null model presented here is just a straightforward extension from an unweighted framework. If one considers the selection of stubs (half-edges), the probability to pick up a stub from node i is $p_i = k_i/2m$ since there are k_i outgoing stubs from node i out of $2m$ stubs in total. The probability to have an edge between i and j is then defined as the probability to select a stub from node i and a stub from node j , so it is proportional to $p_i p_j$ as presented. For a more elaborate development of the null model, see [8].

It follows that if the actual edge weight w_{ij} is larger than the expected edge weight N_{ij} , nodes i and j are strongly connected and assigning them to the same community will lead to a positive modularity gain.

One can see that maximizing the modularity over all possible community partitions (including the number of communities) is a combinatorial problem and cannot be solved in polynomial time, hence the interest for fast algorithms that build an approximation of the optimal modularity.

In the next section, we first review a greedy algorithm, called the Louvain method, to approximate the optimal modularity of a graph and then we will explain the refinements that we brought to this method to segment an input picture.

3 Optimizing the Modularity

There exist different techniques to uncover the community structure in a graph using modularity maximization (a very detailed review has been made by Fortunato [8]). We choose to use a modification of the Louvain method because this method is very successful in terms of computational cost and quality of the detected communities.

3.1 The Louvain Method

The Louvain method introduced by Blondel *et al.* [4] is a greedy algorithm to uncover community structures in a weighted graph using a local decision for each node. This algorithm produces hierarchical communities by recursively aggregating smaller communities.

At the initialization, each node of the graph, i.e. each pixel, defines its own community. At this step, $Q < 0$ since we choose $w_{ii} = 0$ by definition so the modularity is smaller than in the situation where all the nodes are in the same community ($Q = 0$). Then, a node is randomly chosen and each gain of modularity of grouping this node to any of its neighboring communities is computed. The node is finally assigned to the community with the maximal (positive) gain. The procedure iterates over randomly chosen nodes until no positive gain can be found (each node can be selected more than once since a node can be taken out of its pre-assigned community to check if no other community assignment can produce a larger modularity gain). After convergence, all the communities found are aggregated to form a new graph: each community becomes a single node, with a self-loop computed as the sum of the weights of all the internal edges, while the edges between the new nodes are computed as the sum of the old external edges. This two-steps procedure is then recursively applied to each of the aggregated graphs produced until no positive gain can be found for any isolated node (figure 1 illustrates the algorithm on a synthetic graph). Note that optimizing the modularity on each aggregated graph is equivalent to an optimization of the modularity based on the initial graph i.e. the modularity of a community structure on the aggregated graph has exactly the same value as the modularity of the induced underlying communities on the initial graph. This property is very valuable since we want to optimize the community structure of the initial picture-based graph without losing information during the aggregation step.

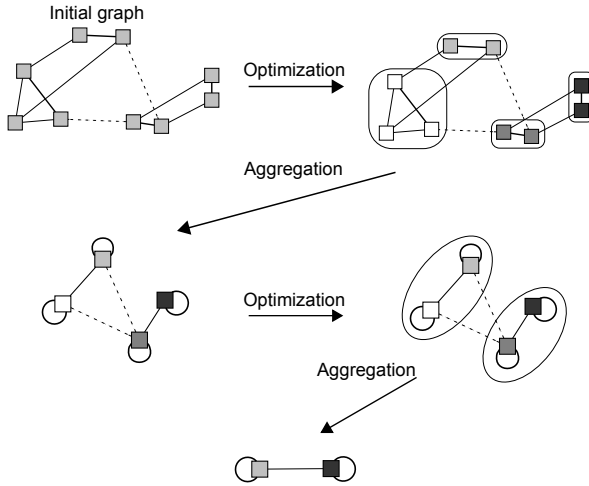


Fig. 1. Representation of the steps applied by the Louvain method on a synthetic graph. At each loop, the algorithm first computes an optimization of the modularity on a graph; then it aggregates the graph according to the communities found.

3.2 A Modified Louvain Method

The Louvain method works well in practical situations but is time consuming at the first iteration (considering the problem of real time video tracking for example) because of the loop over all node. We propose a modification of this algorithm to allow faster computation.

We describe here what we call a basic iteration of the algorithm. First, we compute a gain matrix G that contains the gain of assigning each node with one of its neighbor. This gain matrix is easily computed by

$$G = \frac{1}{2m} \left(2W - \frac{1}{m}kk^T \right) , \tag{4}$$

where k is the vector of degrees. This formulation is very simple because at the beginning of each iteration the communities contain only a single node. Note that the computation of kk^T can be restricted to the indices of non zero elements of W since we are only interested in positive gains.

For each node, we derive an assignment with its best neighbor, defined by the maximum value on each row of the gain matrix G . Then, we consider a directed graph defined by these assignments and we construct the communities as the connected components of this graph. Since the best neighbor assignment can obviously be non-symmetric, the connected components may become very large. This leads to the creation of large communities but it can have the side effect that the negative contribution of the null model becomes larger than the positive contribution of the edge weight, leading to a negative modularity gain. We will

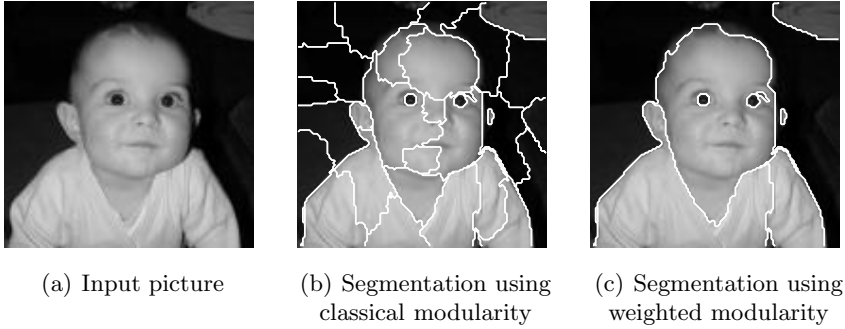


Fig. 2. Segmentation of a baby face (130×132 pixels) using classical and weighted modularity: $\sigma_x = 1.2$, $\sigma_i = 10$, $d_{max} = 3$ and $d_A = 15$. There are 34 regions segmented by the classical modularity and 8 regions segmented by the weighted modularity.

illustrate this phenomenon afterward. In this situation, we recursively remove, from its community, the node with the smallest (negative) gain until the total gain becomes positive.

Finally, once the communities have been defined, they are aggregated to create a new smaller graph.

We iterate this basic iteration until the gain matrix does not contain any positive entry anymore. This algorithm is deterministic and allows, at each step, to assign all the nodes at the same time to a community, leading to a clear improvement in term of computing time.

Fig. 2(a) and 2(b) show the results of the segmentation on a real image using our community detection algorithm. The results of Fig. 2(c) will be discussed later.

One can see that all the regions segmented by the algorithm are coherent but also that the objects of interest are oversegmented (6 regions for the shirt and 7 for the face). This oversegmentation comes from the fact that modularity maximization algorithms cannot yield communities with long chains of nodes. In this context, large regions in the picture (including the background), even with similar intensity levels, cannot be merged together: the distance d_{max} in (1) is small compared to the size of the picture (due to memory limitation), leading to a huge number of disconnected pairs of pixels. Defining a community that contains a large distance between pixels will lead to a smaller (or even negative) modularity because of these disconnected pixels. To illustrate this, suppose that we split the background of an image into 2 adjacent non-overlapping regions r_1 and r_2 and that we denote by ∂r_i the border of width d_{max} of the region i adjacent to region j as depicted in Fig. 3. If we denote by Q_r the contribution in the modularity of a community defined by the region r

$$Q_r = \frac{1}{2m} \sum_{i,j \in r} \left(w_{ij} - \frac{k_i k_j}{2m} \right) ,$$

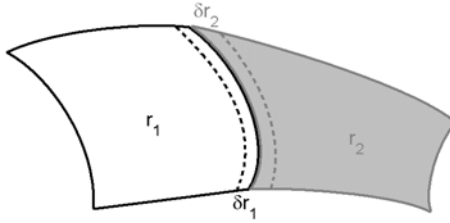


Fig. 3. Representation of 2 adjacent non-overlapping regions with their respective adjacent border

then one can see that

$$Q_{r_1 \cup r_2} = Q_{r_1} + Q_{r_2} + \frac{1}{m} \sum_{\substack{i \in r_1 \\ j \in r_2}} \left(w_{ij} - \frac{k_i k_j}{2m} \right) . \tag{5}$$

The gain of modularity $\Delta Q_{r_1 \leftrightarrow r_2}$ of grouping r_1 and r_2 together is then given by

$$\begin{aligned} \Delta Q_{r_1 \leftrightarrow r_2} &= Q_{r_1 \cup r_2} - Q_{r_1} - Q_{r_2} \\ &= Q_{\partial r_1 \cup \partial r_2} - Q_{\partial r_1} - Q_{\partial r_2} \\ &\quad - \frac{1}{m} \sum_{\substack{i \in r_1 \setminus \partial r_1 \\ j \in r_2}} \frac{k_i k_j}{2m} - \frac{1}{m} \sum_{\substack{i \in \partial r_1 \\ j \in r_2 \setminus \partial r_2}} \frac{k_i k_j}{2m} , \end{aligned} \tag{6}$$

and if the number of pixels in both regions is large, the last two negative terms of this equation dominate, leading to a smaller global modularity.

There are at least two ways to tackle this problem of resolution. First, one can increase the distance d_{max} which in turn increases the size of the border ∂r_i and therefore reduces the number of nodes in $r_i \setminus \partial r_i$. While this is a good theoretical solution that may work on very small images, it is not suitable for typical images; one can check that the number of neighbors for each pixel grows as $O(d_{max}^2)$, making this solution impossible in practice, due to the amount of memory needed to compute W . Another possible solution is to modify the null model (3) in such a way that it takes into account the distance inside a community, defining a weighted version of the modularity. We pursue this idea in the next section.

3.3 The Weighted Modularity

To avoid oversegmentation, the null model should be defined such that it takes into account the fact that an object can be spread over a large area of the picture *i.e.* the computation of the modularity in (2) should be mainly local. In their paper, Reichardt and Bornholdt [17] introduce a constant weighting parameter $\lambda < 1$ in the null model

$$N_{ij} = \lambda \frac{k_i k_j}{2m} ,$$

and show that it produces larger communities. Some other scalings have also been proposed by Delvenne, Yaliraki and Barahona [6]. Based on these observations, we proposed to define the null model as

$$N_{ij} = \Lambda_{ij} \frac{k_i k_j}{2m} , \tag{7}$$

where the matrix Λ is defined as

$$\Lambda_{ij} = \begin{cases} 1 & \text{if } d(i, j) \leq d_\Lambda , \\ 0 & \text{otherwise ,} \end{cases} \tag{8}$$

where d_Λ is a parameter depending on the size of the picture and the size of the objects.

The computation of the aggregation step of the algorithm is straightforward in the context of the classical modularity but it requires some basic matrix computations for the weighted modularity. Let us suppose that the communities are defined by a matrix

$$C \in \{0, 1\}^{p \times n} : C_{ij} = 1 \text{ if community } i \text{ contains node } j ,$$

where p is the number of communities and n the number of pixels. The aggregated graph \tilde{G} is defined by

$$\begin{aligned} \tilde{W} &= C W C^T , \\ \tilde{k} &= \tilde{W} \mathbb{1}_p = C W \mathbb{1}_n = C k , \end{aligned}$$

since we assumed non overlapping regions, thus $C^T \mathbb{1}_p = \mathbb{1}_n$, where $\mathbb{1}_p$ is the vector of size p of all ones.

In the framework of weighted modularity, we still want to keep the property that optimizing the modularity on the initial graph is equivalent to optimizing to modularity of any aggregated graph (up to a constraint). This can still be done if the weight matrix Λ in the aggregated graph is defined by

$$\tilde{\Lambda} = (\tilde{K})^{-1} (C K \Lambda K C^T) (\tilde{K})^{-1} , \tag{9}$$

where K is the diagonal matrix defined by $K = \text{diag}(k)$.

Table 1. Parameters used for the segmentation presented in Fig.4

row 1	$\sigma_x = \sqrt{2} , \sigma_i = 5 , d_{max} = 3 , d_\Lambda = 20$
row 2	$\sigma_x = \sqrt{2} , \sigma_i = 3 , d_{max} = 2 , d_\Lambda = 20$
row 3	$\sigma_x = \sqrt{2} , \sigma_i = 2 , d_{max} = 2 , d_\Lambda = 25$
row 4	$\sigma_x = \sqrt{2} , \sigma_i = 4 , d_{max} = 3 , d_\Lambda = 20$
row 5	$\sigma_x = 2 , \sigma_i = 2 , d_{max} = 4 , d_\Lambda = 30$

One can observe that this definition is consistent with the classical modularity: if we define $\Lambda = \mathbb{1}_n \mathbb{1}_n^T$ then $\tilde{\Lambda} = \mathbb{1}_p \mathbb{1}_p^T$.

The result of the weighted modularity applied to our test picture of the baby face is represented in Fig. 2(c) for $d_\Lambda = 15$. There are 8 regions defined by the algorithm and they have very good visual relevance.

4 Experimental Results

We ran our algorithm on a set of pictures taken from the Berkeley image segmentation database [15] and found relevant contours of objects when the parameter of the method were correctly defined. Some results are displayed in Fig. 4 along with the initial pictures and human segmentation benchmarks. The different pictures are presented in increasing level of complexity according to the benchmark, and the parameters used for each segmentation are presented in table 1. One can see that our algorithm is able to correctly identify the general contour of the objects in each picture but tends to have some difficulties when defining a boundary in low gradient regions like the neck of the camel in row 3. There is also more merging of communities that might be considered, for example in row 4 in the background behind the violinist or in row 5 for the large rock at the right side. Those mergings are not allowed by the algorithm because the communities do not have enough edges between each other but we plan to investigate this question by adding a penalty for the smoothness of the contours.

Fig. 5 shows the result on a picture of elephants. One can see that the algorithm found 7 main regions, uncovering the 2 elephants, and some minor communities with less than 10 nodes due to image quality and artefacts, as shown in Fig. 5(a). Those small communities can be easily postprocessed and merged with the surrounding communities.

The algorithm requires about 2 seconds to segment the 320×480 pictures on a Intel Core 2 Quad, 2.5 GHz with 4 Go ram, using Matlab.

Another important feature of our method is that the detected communities are hierarchical, meaning that each community is only the aggregation of smaller communities. This allows us to extract smaller parts of objects contained in larger areas of the picture. As an example, Fig. 5(b) shows the communities found at the penultimate aggregation step. This step discovers finer boundaries between relevant parts of the picture such as a distinction between the two animals.

The decrease of the number of communities is exponential in the number of basic iterations as shown in Fig. 6, independently of the figure or the number of pixels. Knowing that the assignment on each step is done in $\mathcal{O}(E)$ where E is the number of edges in the graph \tilde{G} , this allows the segmentation of the input image to be done in a reasonable amount of time.

While we observe good segmentation results, we also notice that the communities found can change dramatically with a minor modification of the parameters (σ_x , σ_i , d_{max} or d_Λ). This sensitivity to the parameters is observed

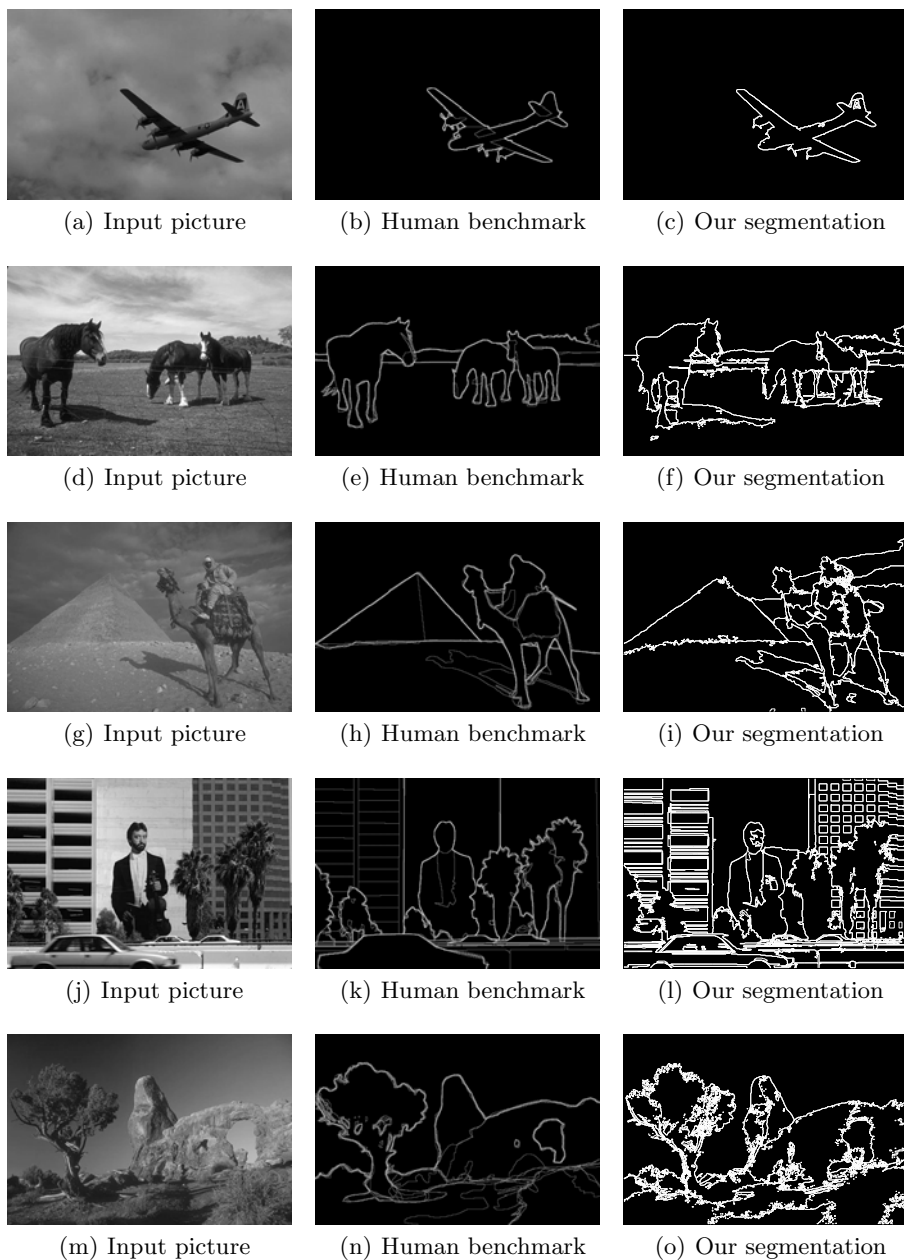
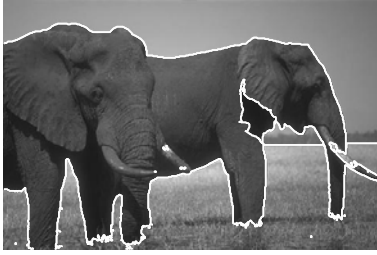


Fig. 4. Segmentation results on some pictures from the Berkeley image segmentation database. Each row presents the initial picture, the human segmentation and the result found by our algorithm.



(a) Segmentation using weighted modularity



(b) Segmentation 1 step before the final aggregation step

Fig. 5. Segmentation of 2 elephants (320×480 pixels) using weighted modularity: $\sigma_x = 5$, $\sigma_i = 9$, $d_{max} = 3$ and $d_A = 25$. In [5\(a\)](#), there are 7 main regions segmented (and some small communities, less than 10 pixels, due to imaging artefact) representing the background and the animals. In [5\(b\)](#), the segmentation at the penultimate aggregation step is displayed showing that there are in fact 2 distinct elephants.

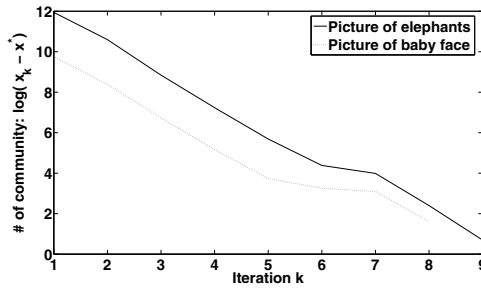


Fig. 6. Evolution of the number of communities showing the exponential decrease in the number of basic iterations

in most segmentation methods but it seems to affect strongly our results. One can then ask if our method is able to derive good parameters based on the modularity value. Unfortunately, it has already been shown by Good *et al.* [\[11\]](#) that the modularity cannot be compared across different graphs. We also observed that the “optimal”¹ modularity value exhibits the shape of a plateau when the parameters vary as represented in [Fig. 7](#). It is not possible to define good parameters leading to a well clustered graph based on this criterion. Because of the high sensitivity of the method to parameters selection, one needs to further investigate this question to better understand the dependencies between parameters and derive good selection rules. This is a topic for future paper.

¹ Optimal in the sense of the best community definition found by the method.

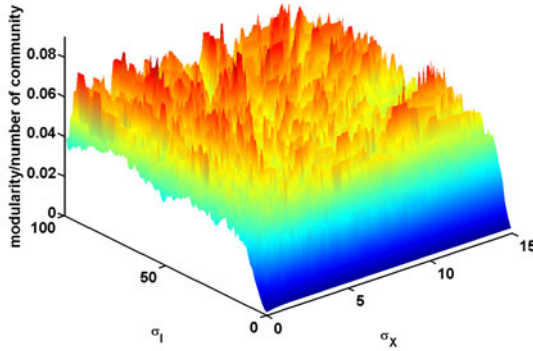


Fig. 7. Modularity, weighted by the number of communities, for different values of the parameters σ_x and σ_i . Since changing the parameters leads to different graphs, the values of modularity cannot be compared.

5 Conclusion

Image segmentation is a difficult problem but we show that when a graph is constructed from a given image using well chosen parameters, extracting communities from this graph can produce salient contour detection of the objects inside the picture. The community definition can be quickly computed by considering an approximation of the Louvain method. This greedy algorithm optimizes a criterion called the modularity that produces relevant community structures.

Classical modularity suffers from a problem of resolution imposing that the radius of communities should stay small. This limitation generally results in oversegmentation of the input picture. We proposed an alternative definition, called weighted modularity, to tackle this problem by weighting the null model according to a smaller radius. Based on this new criterion, we showed that good segmentation can be achieved in a reasonable amount of time. The communities defined are also hierarchical, allowing to uncover smaller entities inside larger objects or shapes.

Parameters selection for the edge weight is a crucial problem and unfortunately, modularity optimization is not able to define the parameters leading to a good pixel clustering, since modularity exhibits the shape of a plateau when it is compared across different weight matrices W for the same image. Another parameter selection can be discussed: as mentioned, a reduction of the oversegmentation can be achieved by using a larger distance radius d_{max} or by considering weighted modularity with a radius d_A . These parameters do not seem to be independent and some work has to be done in this direction to better understand the relationship between them.

An interesting topic for future research is the possibility to use an approximation of the matrix A . At the early step of the method, computing this matrix

can be time consuming. This can be greatly improved by considering a low rank approximation of A if the segmentation results stay accurate.

Finally, we also plan to extend the method to video tracking allowing to track multiple objects on consecutive video frames.

Acknowledgment. This paper presents research results of the Belgian Network DYSCO (Dynamical Systems, Control, and Optimization), funded by the Interuniversity Attraction Poles Program, initiated by the Belgian State, Science Policy Office. The scientific responsibility rests with its authors. This work is also supported in part by “Communauté française de Belgique - Actions de Recherche Concertées”.

Finally, the authors would like to thank S. Chafik and J.C. Delvenne for their contributions on various aspects of this work.

References

1. Alzate, C., Suykens, J.A.K.: Sparse kernel models for spectral clustering using the incomplete cholesky decomposition. In: IJCNN, pp. 3556–3563 (2008)
2. Alzoubi, H., Pan, W.D.: Fast and accurate global motion estimation algorithm using pixel subsampling. *Information Sciences* 178(17), 3415 (2008)
3. Beucher, S.: The watershed transformation applied to image segmentation (1991)
4. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment* 2008(10), 10008 (2008)
5. Cox, I., Rao, S., Zhong, Y.: ”ratio regions”: A technique for image segmentation. In: International Conference on Pattern Recognition, vol. 2, p. 557 (1996)
6. Delvenne, J.C., Yaliraki, S.N., Barahona, M.: Stability of graph communities across time scales. *PNAS* 107(29), 12755–12760 (2010)
7. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. *Int. J. Comput. Vision* 59, 167–181 (2004)
8. Fortunato, S.: Community detection in graphs. *Physics Reports* (2010)
9. Frederix, K., Barel, M.V.: Sparse spectral clustering method based on the incomplete cholesky decomposition. Report TW552, Katholieke Universiteit Leuven (2009)
10. Girvan, M., Newman, M.E.J.: Community structure in social and biological networks. *Proceedings of the National Academy of Sciences of the United States of America* 99(12), 7821–7826 (2002)
11. Good, B.H., de Montjoye, Y.A., Clauset, A.: Performance of modularity maximization in practical contexts. *Physical Review E* 81(4), 046106+ (2010)
12. Ho, J., Lee, K.C., Yang, M.H., Kriegman, D.: Visual tracking using learned linear subspaces. In: Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pp. I–782–I–789 (2004)
13. Olszewska, J.I.: Unified framework for multi-feature active contour. Ph.D. thesis, Universite catholique de Louvain, Ecole polytechnique (2009)
14. Kass, M., Witkin, A., Terzopoulos, D.: Snakes: Active contour models. *International Journal of Computer Vision* 1(4), 321–331 (1988), doi:10.1007/BF00133570

15. Martin, D., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: Proc. 8th Int'l Conf. Computer Vision, vol. 2, pp. 416–423 (2001)
16. Newman, M.E., Girvan, M.: Finding and evaluating community structure in networks. *Physical review. E, Statistical, nonlinear, and soft matter physics* 69(2 Pt 2) (2004)
17. Reichardt, J., Bornholdt, S.: Statistical mechanics of community detection. *Phys. Rev. E. Stat. Nonlin. Soft Matter Phys.* 74(1 Pt 2) (2006)
18. Shi, J., Malik, J.: Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22(8), 888–905 (2000)
19. Sundaramoorthi, G., Mennucci, A., Soatto, S., Yezzi, A.: Tracking deforming objects by filtering and prediction in the space of curves. In: CDC (December 2009)
20. Wang, S., Siskind, J.M.: Image segmentation with minimum mean cut. In: Proceedings of the Eighth IEEE International Conference on Computer Vision, ICCV 2001, vol. 1, pp. 517–524 (2001)
21. Werman, M., Peleg, S., Rosenfeld, A.: A distance metric for multidimensional histograms. *Computer Vision, Graphics, and Image Processing* 32(3), 328–336 (1985)
22. Wu, Z., Leahy, R.: An optimal graph theoretic approach to data clustering: theory and its application to image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 15(11), 1101–1113 (2002)

BCIF: Another Algorithm for Lossless True Color Image Compression

Stefano Brocchi and Elena Barcucci

Dipartimento di Sistemi e Informatica, Università di Firenze
Viale Morgagni, 65 - 50134 Firenze - Italy
{brocchi,barcucci}@dsi.unifi.it

Abstract. In this paper we present an algorithm for the lossless compression of true color images. Our aim was to develop a practical algorithm with a fast decompression phase. The algorithm executes a block adaptive predictive filtering phase, followed by a color filtering phase that exploits color correlation, and finally compresses the prediction errors through context assignment and Huffman coding. Comparing the proposed algorithm with competing standards as Jpeg2000 and Jpeg-LS, we show how our method yields better compression ratios without having a slower decompression speed.

Keywords: Lossless compression, predictive coding, Huffman codes.

1 Introduction

Much recent effort has been spent to develop methods for efficient lossless image encoding. Even if in many contexts a lossy compression is a reasonable choice, there are many cases where it would be inappropriate, as for example in biomedical imaging.

Of the many existing lossless compressed image formats, the most used is the Portable Network Graphics (PNG) standard [9]. In spite of its diffusion, the underlying algorithm has been outperformed by many more recent methods, able to obtain a greater compression ratio using fewer resources.

Many developed algorithms aim to obtain the highest possible level of compression without sparing on resources. Even achieving impressive compression ratios, their very large time requirements represent a very heavy limitation to their practical use; some notorious algorithms in this family are MRP ([3], [2]), TMW [4] and CALIC-A ([8]).

On the other hand, there is another family of algorithms that obtain a good compression ratio while maintaining a low complexity. Two of the most popular and efficient are the LOCO algorithm, evolved in the Jpeg-LS standard [7], and the Jpeg 2000 standard [10]. Both built for lossy and lossless compression, these techniques allow the generation of files in average a 15% bigger than the state of art technique MRP (see benchmarks in [3]) while being hundreds of times

faster during encoding. Other highly effective algorithms have a better trade-off between time and compression ratio than MRP, but the Jpeg standards are still considered the ones obtaining the best compromise between these two factors.

In this work we propose a new lossless image compression method, specialized for true color images. The procedure is optimized for contexts where the decompression process is applied many more times than the compression one; this covers a huge part of the usage cases, as typically an image is encoded once, and after its storage on a disk it must be decompressed every time a user wants to visualize it. We created an algorithm with a superior compression ratio to the Jpeg standards, and with a decompression phase fast as Jpeg-LS and faster than Jpeg-2000. Being an asymmetrical algorithm, the compression phase is several times slower than the decoding one, but it is still contained in acceptable bounds, making the proposed algorithm superior in the considered context.

Finally, the algorithm is released as free software, making it a good alternative to patented software.

2 The Algorithm

The proposed algorithm is an evolution of the PCIF algorithm described in [1]. The first two stages involving filtering and color filtering are applied similarly, but their efficiency has been improved thanks to the usage of new color filters and to a more complex filter selection heuristic. With this new approach, the complexity of the filter determination phase is higher, but there is an improvement of the compression ratios that does not alter the decompression speed. The compression procedure represents instead the main difference between the proposed algorithm and the previous PCIF method, that at this point decomposed the image in bitplanes in order to compress them independently. By compressing the prediction errors without this decomposition, this new method allows both a faster and a more effective compression, even if it does not allow a simple parallelization as in PCIF.

The current description and implementation of the proposed algorithm focus on 24 bit depth color images (8 bits for each one of the RGB components), but the algorithm could be easily extended to color images with different bit depths.

In its first stage, the algorithm applies a block adaptive spatial filtering process. Every 8×8 zone of the image is object of a filtering function selected for that particular zone, chosen from a set of 13 predictive functions; the value of each sample is replaced with its difference from the prediction for that point, module 256. In order to maintain a low complexity, only adjacent pixels are chosen to predict a given sample, and the operations involved in the filtering functions are mainly sums and bit shifts. As usual in predictive image filtering, the predictive functions are built upon assumptions of smoothness of the image.

In the second stage the algorithm applies a block adaptive color filtering phase. Again every sample is replaced with its difference with a predicted value, but such predictions are computed in function of values relative to the same pixel, but to different color planes. By executing color decorrelation *after* the spatial

filtering phase, the algorithm is able to apply to the image a block adaptive color transform without compromising the assumptions of smoothness necessary in the previous stage; as discussed in [1], this is an innovative approach that yields better results in comparison to the standard solution of applying an unique color transform as a preprocessing stage, as in the Jpeg algorithms.

In the third stage the algorithm proceeds in compressing the prediction errors through the assignment of a context to every different point, each of them associated to a different set of Huffman codes. The context of a point is determined in function of the adjacent prediction errors and of a parameter k , computed recursively, related to the estimated variance of the values in the given context. Since there is a need to use a different Huffman tree for every context, we defined an efficient encoding of the Huffman trees themselves, allowing the compression of these structures in very few bits.

In figure 1 there is depicted a flow diagram representing the whole algorithm in its encoding and decoding phases. While the compression procedure is actually implemented in separate phases, the operations necessary for decompression are done during an unique pass on the image, allowing a fast decoding procedure and the streaming of the image while the decoding process is being executed.

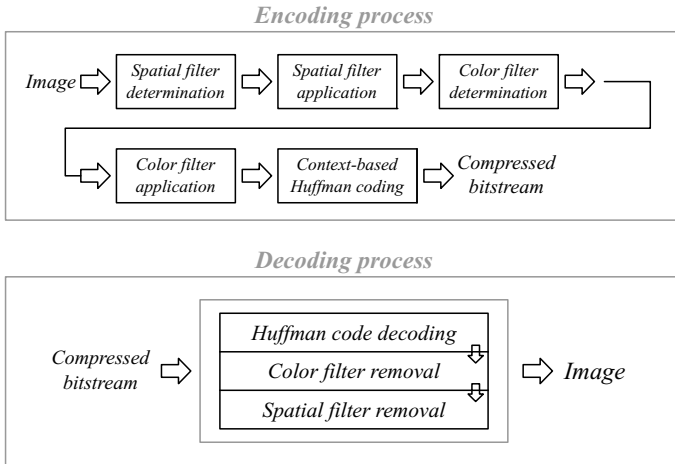


Fig. 1. Flow diagram of the encoding and decoding procedure

3 Filtering

In this first filtering phase the image is scanned starting from the upper rightmost pixel and proceeding from right to left and then from high to low. The predictions are computed for a given sample x in function of the adjacent values a, b, c and d depicted in figure 2. For each 8×8 zone of the image, a different predictive function is selected between the following:

A	X	
C	B	D

Fig. 2. Samples used for filtering

1. $f(x) = 0$
2. $f(x) = a$
3. $f(x) = b$
4. $f(x) = c$
5. $f(x) = d$
6. $f(x) = a + (b - c)/2$
7. $f(x) = b + (a - c)/2$
8. $f(x) = (a + b)/2$
9. $f(x) = (a + b + c + d + 1)/4$
10. $f(x) = (a + d)/2$
11. $f(x) = a + b - c$ if $0 \leq a + b - c < 256$
 $f(x) = 0$ if $a + b - c < 0$
 $f(x) = 255$ if $a + b - c > 255$
12. $f(x)$ defined by the following:

If $a \leq c \leq b$, $f(x) = a + b - c$,
 otherwise $f(x)$ is equal to the Paeth filter, defined as follows:

```

p = a + b - c;
pa = abs(p - a);
pb = abs(p - b);
pc = abs(p - c);
if pa <= pb and pa <= pc then f(x) = a
  else if pb <= pc then f(x) = b
    else f(x) = c;
```

All divisions are intended as integer divisions, allowing them to be computed as inexpensive bit shifts. The functions are the same of those used in [1], except for functions 5, 9 and 10 that use the point d instead of the point to the upper left of x , allowing the streaming of the image per rows during decompression.

4 Color Filtering

After the spatial filtering, the algorithm applies a color filtering phase where there is an attempt to reduce the prediction errors by applying a further filtering phase that exploits the correlation between color planes. This has a strong impact on the compression ratio as zones with sharp edges usually cause high prediction errors in all three color planes. In these cases, one of the high prediction errors

will be used to predict the others, and if the three values are similar then two of them will result near to zero after this operation.

In order to maintain a low complexity, to estimate a prediction error only values relative to the same pixel are used; for the same reason, there are only 6 possible color filtering functions. Said r, g and b the color components of a pixel (respectively: red, green and blue), the predictive functions are the following:

1. No filter
2. $g = g - b$;
 $r = r - g$;
3. $g = g - b$;
 $r = r - b$;
4. $g = g - r$;
 $b = b - r$;
5. $g = g - r$;
 $b = b - g$;
6. $b = b - g$;
 $r = r - g$;

We can observe that for every color filtering function h , $h(r, g, b)$ can be easily inverted in order to rebuild the original triplet of values. As for spatial filters, the selected functions are encoded in the compressed bitstream using Huffman codes.

We have depicted in figure 3 the statistics about the filters chosen for the Kodak and Waterloo test image sets described hereafter. For each filter, it is shown how many times it was selected as optimal for a zone, in proportion to the total number of zones. These ratios vary greatly from image to image, hence even filters that are applied a limited number of times may have given a significative contribution to some specific images.

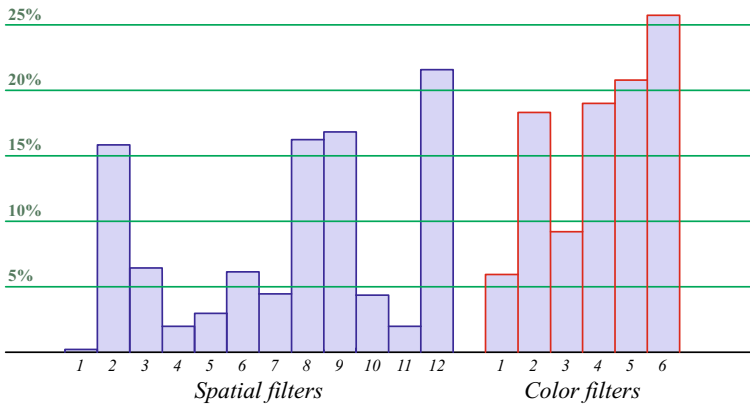


Fig. 3. Statistics on the selected filters and color filters

5 The Filter Selection Model

In order to select the optimal predictive function for every zone, the algorithm estimates the encoding cost of the possible resulting prediction errors. Such cost is computed by considering a first order entropy model based on the frequency of the prediction errors in the previously filtered areas of the image. The algorithm computes a better estimation for the first zones of the image, and further on, having gathered more data about the prediction errors, it settles for a faster procedure. The described heuristics are applied both for spatial filters and for color filtering functions.

We chose to use as a metric the first order entropy of the prediction errors mainly for two reasons; as a first one, this keeps the encoding complexity low allowing a good compression time. As a second motivation, since several non trivial operations are done on the image before encoding, it is not easy to determine a more complex model that correctly determines optimality. In fact, we did some experiments using also more complex models, but this did not improve the performances. Anyway increasing compression complexity to improve the compression ratio without compromising the decompression speed could be an interesting development, and studies about a better heuristic for filter selection may be an interesting object for future work.

For a generic zone i in the first 10 zones of the image, the algorithm computes for every coding function f the first order entropy of the samples in $1 \dots i - 1$ together with $f(i)$; the function that gives the minor coding cost is chosen. Starting from zone 11, the cost of every prediction error is estimated as its encoding cost, in bits, in a first order entropy model relative to the previous zones. Such costs are updated at zones of index 10, 20, 40, 80, \dots , 1280, and beyond 1280 every 1000 zones.

The goal of this technique is to take into account, in the initial areas of the image, the prediction errors in all the previous zones together with the newly generated ones in the considered zone. With this method, the algorithm obtains very good estimations at the beginning of the encoding. On the other hand, when the compression process has advanced and many statistics on the image have already been gathered, the procedure settles for a weaker approximation in order to improve the encoding speed.

The information about which filters have been chosen are compressed through Huffman codes and are encoded in the compressed file.

6 Compression

After the two filtering phases, the resulting prediction errors have a reduced correlation between adjacent values; hence to obtain a compression ratio noticeably better than the first order entropy, it is necessary to consider information deriving from a large number of samples. A straightforward method would be to use a high order arithmetic coding, possibly on the style of PPM methods (see for example [6]), but this would compromise time performances, in contrast with our goal to create a fast decompressing algorithm.

We propose a solution to this problem that allows a good compression ratio maintaining a fast decompression phase. The samples are encoded in function of an assigned context, each one of them having a different set of Huffman codes. This operation aims to create some contexts with high entropy and others with a very small variance and lots of values close to 0; this exploits at most the advantage of having different Huffman codes for each context.

In order to determine the context of a prediction error, the algorithm assigns to each one of these a value $I(x, y, c)$ related to the estimated variance of the current context; the parameters x and y represent the horizontal and vertical coordinates of the sample, while c is the color band. Such parameter is computed recursively in function of the previous values of I , in order to consider information deriving from many previous samples with only a limited number of operations. The assignment of a context is then done in function of such values.

The image is encoded starting from the lower left (point $(0, 0, 0)$) and proceeding from left to right and then from low to high. The following function defines the pseudo code for the computation of $I(x, y, c)$, i.e. the context of the point of coordinates (x, y, c) . The matrix M represents the prediction errors obtained after the color filtering phase, and I contains the contexts assigned to the samples to be encoded. Note that the procedure must use, for the computation of $I(x, y, c)$, only the values of I that proceed (x, y, c) in the scan order.

```

FUNCTION assign_context(x, y, c)
sum = abs(M[x - 1][y][c]) + abs(M[x][y - 1][c]);
if (sum == 0) {
    chaos = 0;
} else {
    chaos = min(log2(sum * 2) + 1, 7);
}
isum = I[x - 1][y][c] + I[x][y - 1][c];
if (c > 0) {
    chaos2 = I[x][y][c - 1];
} else {
    chaos2 = isum / 2;
}
curChaos = chaos + (isum + chaos + chaos2) / 4;
I[x][y][c] = (curChaos / 2);
return curChaos + c * 16;

```

Every division is intended as an integer division and can hence be computed as a bit shift; the \log_2 function returns the integer part of the base-2 logarithm of the argument. The function `abs` is the absolute value function, and it is used as the prediction errors are represented in the interval $[-128, 127]$. We do not describe the special behaviors defined for the various border cases.

To obtain an efficient implementation, the matrices M and I are never entirely represented: since the encoding and decoding procedures always require only rows x and $x - 1$, only the arrays representing these two lines are actually kept in memory.

Many factors have been considered for the creation of this function, not last an extensive experimentation. The adjacent values weight strongly on the definition of a context, and since variations on the sum of their absolute values seem to be much more significative near 0, we considered their logarithm. The values in I represent an information that derives from a larger context, where, thanks to the definition of I , the samples are considered exponentially less with the growth of their manhattan distance from the point to be encoded. Finally, note that since for every point $curChaos < 16$, thanks to the last instruction values in different color planes are always designed to different contexts.

Several other refinements have been considered to improve the compression ratio:

- In contexts where the frequency of zeros is higher than 30%, the zeros are run length encoded before compression. With this method we avoid the negative consequences of using an integer number of bits per symbol.
- In this case the values immediately after the symbol 0 are encoded with another 'auxiliary' Huffman tree, allowing the usage of a higher order entropy model for these special cases.

7 Huffman Tree Coding

Since the algorithm requires a noticeable number of Huffman trees that must be embedded in the compressed file, we have developed an efficient encoding for these structures. This technique is based on the observation that the only information required to build the optimal codes for a set of symbols is the lengths of the codes themselves; being able to encode only this information in the encoded bitstream instead of the entire tree structure yields a noticeable improvement in the compression ratio. We now describe how an Huffman tree can be built starting from the bit lengths of the encoded symbols; in the following we show how to assure that the codes used during the encoding and decoding phases match.

Let's suppose to have as input a set of positive integers representing the length, in bits, of the optimal codes for considered alphabet. Wanting to rebuild a corresponding tree, we can use an algorithm that mirrors the classical one to construct Huffman trees from the frequencies of the symbols; as a first step, consider a set of trees having only one terminal node, each one of weight equal to the bit length of a different symbol. At each step, merge the two trees with the maximal weight w , and assigns to the resulting tree weight $w - 1$. It is easy to verify that for initial weights corresponding to Huffman codes, at each step there are always two trees with equal maximal length. Iterating the procedure until there is only one tree left, we obtain an Huffman tree where each symbol has a code with the desired length.

Being able to rebuild an Huffman tree from an array of bit lengths, we could encode the tree very efficiently in a compressed bitstream by simply storing this array, containing only small positive integers. There is no guarantee anyway that

the decompression procedure could be able to rebuild exactly the original tree, and not another one with a similar structure but with different codes for the symbols. To solve this problem, the algorithm executes the following steps:

- During encoding, in a first pass compute the frequencies of the symbols
- Use these frequencies to build an Huffman tree defining the optimal bit length of every symbol
- Extract the bit lengths of every symbol, and use the described reconstruction procedure to build another optimal Huffman tree from these values
- Encode the symbols in the compressed bitstream using this new tree
- Encode in the compressed file the array of bit lengths

The compressor will then be able to rebuild exactly the same Huffman tree during decompression, assuming the reconstruction algorithm is deterministic (i.e. solves in any deterministic way a tie between weights in more than two nodes). This property has no impact on the compression ratio, as any fixed criteria to select which two trees with equal weight are merged will construct a structure that is optimal in respect to the initial frequencies.

Considering further assumptions on the frequency of the symbols, the algorithm is able to optimize even more the encoding of the array. Since the difference in frequency of two consecutive values is usually small, we expect the bit lengths of their codes to be very close; for this reason, the algorithm encodes for each value only its difference from the previous one. For such differences, the algorithm makes use of Golomb-Rice style codes preceded by a sign bit. Thanks to such differential coding, the compressed filesize is decreased by approximately 10-15 KB for a generic image, which can be a noticeable amount if the image is small.

8 Benchmarks

The compression ratios of the algorithm have been compared to four other lossless formats: the pcif format¹ [1], the png format [9], the lossless mode of the Jpeg2000 standard² [10] and the loco algorithm used in the Jpeg-LS standard³ [7]. The results can be seen in figure 4; we have named the proposed algorithm bcif. Details about the performances on the single images of the first two classical image sets are in table 1, where the column labeled bmp represents the uncompressed file sizes. All benchmarks are also available online⁴.

Two classical image sets used for benchmarking are the well known Kodak image set⁵, containing 24 photographic images of size 768×512 or 512×768 , and

¹ The implementation is available at www.researchandtechnology.net/pcif/

² As implemented on www.kakadusoftware.com

³ As available at the reference implementation at <http://www.hpl.hp.com/loco/>; since it does not include a color transform, we implemented the one suggested in the original paper [7] to allow the loco algorithm to be competitive.

⁴ <http://www.researchandtechnology.net/bcif/benchmarks.php>

⁵ Available at <http://www.r0k.us/graphics/kodak/>

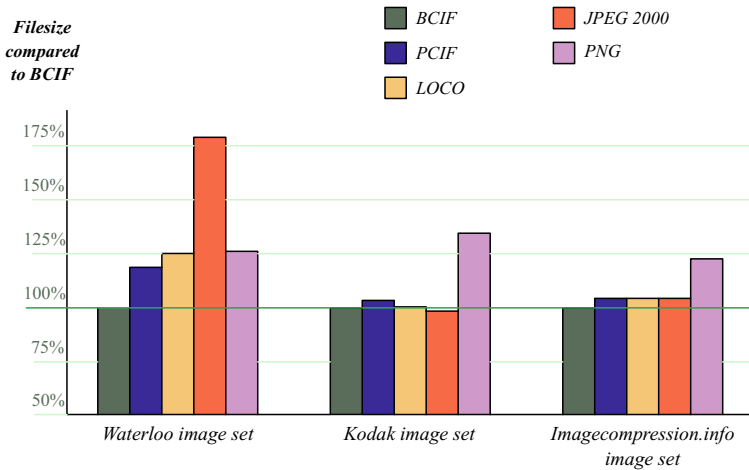


Fig. 4. Comparative benchmarks on the three image sets

the Waterloo image set⁶, with images ranging from dimensions of 512×512 to 1118×1105 and containing both photographic images (lena, monarch, peppers, sail, tulips) and artificial computer generated images (clegg, frymire, serrano). We have also considered a recently proposed image set⁷ containing several high resolution images, in order to experiment the efficacy of the algorithm on large amounts of data. We find that this last image is interesting as it may also be similar, for what regards size and properties, to typical images that photographic experts may have to handle. We call this set the imagecompression.info image set, or I.C.I. image set for short.

We can observe from the tables that the bcif algorithm is an optimal evolution of the pcif algorithm, since it is both superior in compression ratio and faster thanks to its simpler coding procedure. In comparison to the Jpeg standards, the bcif algorithm obtains similar compression ratios for photographic images and yields much better results for artificial ones. The bcif compression ratio is superior in comparison to both of these standards, producing an overall filesize noticeably smaller than both Jpeg-LS and Jpeg2000.

The decompression time of the proposed algorithm has been measured and compared to the one of Jpeg-LS; each time has been computed as an average of 5 decompressions executed on an AMD Athlon 2600 processor. As shown in table 2, the sums of the decompression times on the Waterloo and Kodak benchmark images results to be very similar. The only images where the BCIF algorithm takes noticeably more than the LOCO algorithm are those where the compression ratio of the first is much greater than the one of the latter.

⁶ Available at <http://links.uwaterloo.ca/Repository.html>

⁷ Available at http://www.imagecompression.info/test_images/

Table 1. Compression results

Filename \ size (KB)	bmp	bcif	pcif	loco	jp2	png
Kodak 01	1152	506	516	511	498	719
Kodak 02	1152	452	467	450	439	603
Kodak 03	1152	380	398	374	388	491
Kodak 04	1152	453	463	456	437	622
Kodak 05	1152	535	559	547	519	767
Kodak 06	1152	462	477	465	460	604
Kodak 07	1152	404	426	404	408	553
Kodak 08	1152	534	553	554	534	769
Kodak 09	1152	431	443	437	427	569
Kodak 10	1152	438	450	441	434	579
Kodak 11	1152	452	464	448	446	606
Kodak 12	1152	412	420	402	415	518
Kodak 13	1152	591	604	601	569	803
Kodak 14	1152	506	520	501	487	675
Kodak 15	1152	424	443	426	431	598
Kodak 16	1152	421	432	416	421	521
Kodak 17	1152	442	454	437	435	587
Kodak 18	1152	558	565	562	516	762
Kodak 19	1152	479	490	482	463	655
Kodak 20	1152	350	375	362	387	480
Kodak 21	1152	484	495	481	468	622
Kodak 22	1152	513	524	517	483	685
Kodak 23	1152	419	434	417	407	544
Kodak 24	1152	483	508	493	488	689
clegg	2100	378	571	646	1369	490
frymire	3621	413	641	806	1560	361
lena	768	414	431	442	434	500
monarch	1152	450	468	448	431	621
peppers	768	331	343	327	327	441
sail	1152	541	554	544	511	777
serrano	1463	152	242	282	623	147
tulips	1152	500	527	508	477	691
Total	39827	14324	15274	15201	16706	19064

Since the proposed algorithm obtains greater compression ratios than Jpeg-LS and has approximately the same decoding speed, it should be preferable in every case where the decompression phase is executed most often. The Jpeg 2000 format has not been benchmarked as it is known that it results to be several times slower than Jpeg-LS as reported in [5], and hence we do not expect it to be competitive.

On the other hand, the compression time of the proposed algorithm is actually about 5 times greater than the decompression time, while Jpeg-LS, being symmetrical, requires approximately the same time than in decoding. Even if the

Table 2. Decompression time

Filename \ time (ms)	LOCO decode	BCIF decode	LOCO encode	BCIF encode
Kodak 01	166	156	148	797
Kodak 02	160	148	148	783
Kodak 03	146	168	134	776
Kodak 04	162	156	142	793
Kodak 05	170	152	150	797
Kodak 06	158	158	142	793
Kodak 07	154	154	136	793
Kodak 08	174	154	154	803
Kodak 09	160	154	144	797
Kodak 10	158	156	146	799
Kodak 11	154	150	140	781
Kodak 12	154	150	138	791
Kodak 13	172	160	154	795
Kodak 14	168	152	148	799
Kodak 15	156	152	138	785
Kodak 16	154	148	138	789
Kodak 17	154	146	144	783
Kodak 18	172	140	154	779
Kodak 19	162	144	148	781
Kodak 20	130	138	114	753
Kodak 21	168	150	148	803
Kodak 22	170	154	154	783
Kodak 23	160	158	142	793
Kodak 24	160	172	148	795
clegg	264	284	224	1317
frymire	280	456	198	2135
lena	124	94	104	542
monarch	164	154	144	797
peppers	106	96	100	532
sail	176	144	158	781
serrano	96	194	82	905
tulips	172	146	150	787
Total	5224	5238	4612	26723

bcif compression time is still acceptable, this makes Jpeg-LS be a more feasible algorithm for cases where the encoding must be executed a number of times comparable to those when decoding is needed.

9 Conclusions

Making use of the combination of several different methods, the proposed algorithm obtained a competitive decompression time and bested in compression ratio the practical algorithms that have actually become a standard. While some

of the used techniques as spatial filtering are well known, other stages of the encoding procedure are new and deserve further studies, as the method for color decorrelation and the approach for context modeling and encoding.

The implementation of the algorithm along with its source code is available at www.researchandtechnology.net/bcif/

References

1. Barucci, E., Brlek, S., Brocchi, S.: PCIF: An algorithm for lossless true color image compression. In: Wiederhold, P., Barneva, R.P. (eds.) IWCIA 2009. LNCS, vol. 5852, pp. 224–237. Springer, Heidelberg (2009)
2. Matsuda, I., Kaneko, T., Minezawa, A., Itoh, S.: Lossless coding of color images using block-adaptive inter-color prediction. In: IEEE International Conference on Image Processing (ICIP 2007), vol. 2, pp. 329–332 (2007)
3. Matsuda, I., Ozaki, N., Umezumi, Y., Itoh, S.: Lossless coding using variable block-size adaptive prediction optimized for each image. In: Proceedings of 13th European Signal Processing Conference, WedAmPO3 (2005)
4. Meyer, B., Tischer, P.: TMW - a new method for lossless image compression. In: Proceedings of the 1997 International Picture Coding Symposium (PCS 1997), pp. 533–538 (1997)
5. Santa-Cruz, D., Grosbois, R., Ebrahimi, T.: JPEG 2000 performance evaluation and assessment. *Signal Processing: Image Communication* 17(1), 113–130 (2002)
6. Shkarin, D.: PPM: one step to practicality. In: Storer, J.A., Cohn, M. (eds.) Proceedings of the IEEE Data Compression Conference 2002, Snowbird, Utah, pp. 202–211 (2002)
7. Weinberger, M.J., Seroussi, G.: The LOCO-I lossless image compression algorithm: principles and standardization into JPEG-LS. *IEEE Transactions of Image Processing* 9(8), 1309 (2000)
8. Wu, X., Memon, N.: Context-based, adaptive, lossless image coding. *IEEE Transactions on Communications* 45(4) (1997)
9. ISO/IEC 15948, Portable network graphics (PNG) specification, W3C Recommendation (2003)
10. ISO/IEC JTC 1/SC 29/WG 1, ISO/IEC FCD 15444-1, Information technology - JPEG 2000 image coding system (March 2000)

Distance Measures between Digital Fuzzy Objects and Their Applicability in Image Processing

Vladimir Ćurić¹, Joakim Lindblad², and Nataša Sladoje³

¹ Centre for Image Analysis, Uppsala University, Sweden
vlada@cb.uu.se

² Centre for Image Analysis, Swedish University of Agricultural Sciences,
Uppsala, Sweden and
Mathematical Institute, Serbian Academy of Sciences and Arts, Belgrade, Serbia
joakim@cb.uu.se

³ Faculty of Technical Sciences, University of Novi Sad, Serbia
sladoje@uns.ac.rs

Abstract. We present two different extensions of the Sum of minimal distances and the Complement weighted sum of minimal distances to distances between fuzzy sets. We evaluate to what extent the proposed distances show monotonic behavior with respect to increasing translation and rotation of digital objects, in noise free, as well as in noisy conditions. Tests show that one of the extension approaches leads to distances exhibiting very good performance. Furthermore, we evaluate distance based classification of crisp and fuzzy representations of objects at a range of resolutions. We conclude that the proposed distances are able to utilize the additional information available in a fuzzy representation, thereby leading to improved performance of related image processing tasks.

Keywords: Fuzzy sets, set distance, registration, classification.

1 Introduction

Distances between sets are useful for many different applications in image processing, for instance, object matching [3, 8], image registration [12] and image retrieval [11]. However, it is a challenging task to differentiate between crisp discrete representations of similar objects if the spatial resolution is insufficient. An example of such a situation can be seen in Fig. 1, where a crisp discrete representation of a disk, Fig. 1B, a crisp discrete representation of an octagon, Fig. 1E, are difficult to visually associate with the correct type of continuous shape (disk and octagon).

In recent years, fuzzy approaches have gained increased popularity in image processing. In [16, 17, 18] it is shown that a fuzzy representation provides higher precision and accuracy of different feature estimates than a crisp one. Keeping this fact in mind, it can be assumed that distances between fuzzy objects representations provide better discriminatory power than distances between crisp

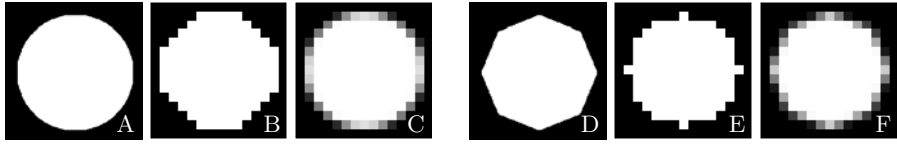


Fig. 1. A: Continuous crisp disk, B: Crisp discrete representation of a continuous disk (obtained by Gauss centre point digitization), C: Fuzzy discrete representation of a continuous disk (obtained by coverage digitization), D: Continuous crisp octagon, E: Crisp discrete representation of a continuous octagon, F: Fuzzy discrete representation of a continuous octagon

representations at the same resolution. This assumption is supported by the fact that a fuzzy discrete representation of an object (see Fig. 1C and Fig. 1F) often appears visually more similar to the corresponding continuous object, than a crisp representation at the same resolution. These observations motivate the study on distance measures between fuzzy sets presented in this paper.

It is recently shown that the Sum of minimal distances and the Complement weighted sum of minimal distances have good performances for binary image registration [12,7]. We extend these distances to distances between fuzzy sets by following two different approaches. Having on mind matching and registration as possible applications, we evaluate distance measures with respect to several criteria of importance for such applications. We study whether the distances are monotonically increasing with respect to increasing translation and rotation of the object. Additionally, we investigate noise sensitivity of the observed distances. To further evaluate the proposed approach, we compare the correct classification rates, when utilizing the best performing fuzzy and crisp set distances for discriminating discrete representations of disks and octagons at a range of resolutions.

2 Background

2.1 Basic Notions

A fuzzy set \mathcal{S} on a reference set X , is a set of ordered pairs $\mathcal{S} = \{(x, \mu_{\mathcal{S}}(x)) : x \in X\}$, where $\mu_{\mathcal{S}} : X \rightarrow [0, 1]$ is the membership function of the fuzzy set \mathcal{S} , [19]. Many concepts of fuzzy sets are based on α -cuts. An α -cut of a fuzzy set \mathcal{S} , is the set ${}^{\alpha}\mathcal{S} = \{x \in X : \mu_{\mathcal{S}}(x) \geq \alpha\}$, $\alpha \in (0, 1]$. Height of a fuzzy set \mathcal{S} is $h(\mathcal{S}) = \max_{x \in X} \mu_{\mathcal{S}}(x)$, while the support of \mathcal{S} is defined as $Supp(\mathcal{S}) = \{x \in X : \mu_{\mathcal{S}}(x) > 0\}$. The complement $\overline{\mathcal{S}}$ of a fuzzy set \mathcal{S} , is $\overline{\mathcal{S}} = \{(x, 1 - \mu_{\mathcal{S}}(x)) : x \in X\}$.

The fuzzy representation of objects used in this paper is pixel coverage representation [16], where membership of a pixel is equal to the relative area of the pixel that is covered by the object. More formally, for a given continuous object $O \subset \mathbb{R}^2$, inscribed into an integer grid with pixels $p_{(i,j)}$, the n -level quantized pixel coverage digitization of the object O is

$$\left\{ \left((i, j), \frac{1}{n} \left\lfloor n \frac{A(p_{(i,j)} \cap O)}{A(p_{(i,j)})} + \frac{1}{2} \right\rfloor \right) : (i, j) \in \mathbb{Z}^2 \right\}, \tag{1}$$

where $A(X)$ denotes the area of the set X , and $\lfloor x \rfloor$ denotes the largest integer which is not greater than x . The set $\{0, \frac{1}{n}, \dots, \frac{n-1}{n}, 1\}$ represents the pixel coverage values for n -level quantized pixel coverage digitization. This set corresponds to the set of non-zero membership levels, e.g., $n = 1$ for a binary images, while $n = 255$ provides the set of membership levels for an 8-bit image. We use this representation since it has been shown to provide higher precision and accuracy of different feature estimates than the crisp object representation [16, 17]. It is also corresponds well with the outcome of many imaging situations. However, the applicability of the methods presented in this study is not restricted to this type of fuzzy representations.

2.2 Related Work on Distances between Crisp Sets

Distances between two crisp sets of points $A, B \subset \mathbb{Z}^n$, $A, B \neq \emptyset$, are mostly based on the point-to-set distance; the point-to-set distance between point a and set B is defined as

$$d(a, B) = \inf_{b \in B} d(a, b). \tag{2}$$

For the underlying point-to-point distance $d(a, b)$, we use the Euclidean distance, i.e., $d(a, b) = \|a - b\|_2$.

The first proposed distance between two sets A and B , and widely used in different applications, is the Hausdorff distance, d_H , defined as

$$d_H(A, B) = \max(\sup_{a \in A} d(a, B), \sup_{b \in B} d(b, A)). \tag{3}$$

The Hausdorff distance is highly dependent on two points from the observed sets and, hence, sensitive to outliers. Several modifications of the Hausdorff distance are introduced to reduce the influence of outliers to the distance measure [8].

In [12] it is suggested that a distance measure applicable for image registration related problems should simultaneously: (i) utilize all points from the set; (ii) consider spatial position of the points and the sets. One distance that fulfills these conditions is the Sum of minimal distances, proposed in [9],

$$d_{SMD}(A, B) = \frac{1}{2} \left(\sum_{a \in A} d(a, B) + \sum_{b \in B} d(b, A) \right). \tag{4}$$

This distance has good performance for image registration [12]. The Sum of minimal distances is investigated further and the Complement weighted sum of minimal distances, d_{CW} , is proposed in [7]. In d_{CW} each point in the set is weighted by the distance to the complement of the set

$$d_{CW}(A, B) = \frac{1}{2} \left(\frac{\sum_{a \in A} d(a, B) \cdot d(a, \bar{A})}{\sum_{a \in A} d(a, \bar{A})} + \frac{\sum_{b \in B} d(b, A) \cdot d(b, \bar{B})}{\sum_{b \in B} d(b, \bar{B})} \right). \tag{5}$$

Based on the empirical evaluation in [7], it is concluded that the Complement weighted sum of minimal distances, among the considered distance measures, has the best performance for binary image registration.

2.3 Related Work on Distances between Fuzzy Sets

There exist a number of different distances defined for fuzzy sets. A good overview on distance measures between fuzzy sets can be found in [2]. Some of the measures provide a fuzzy number as output [15]. However, we consider only distances between fuzzy sets that provide crisp distance values; for the observed application (object matching) it is not obvious how to use distance values represented by fuzzy numbers.

There exist several different possibilities how to extend a crisp distance to a fuzzy one [1]. The most common approach is to use integration over α -cuts [14]. Integration over α -cuts is a general principle for extension of properties and relations on crisp sets to corresponding ones on fuzzy sets. A distance measure between two fuzzy sets \mathcal{A} and \mathcal{B} can be defined by integration over all α -cuts,

$$d^\alpha(\mathcal{A}, \mathcal{B}) = \int_0^1 d({}^\alpha A, {}^\alpha B) d\alpha, \tag{6}$$

where d is a crisp set distance. For example, the Hausdorff distance between two fuzzy sets \mathcal{A} and \mathcal{B} is defined as, [14]

$$d_H^\alpha(\mathcal{A}, \mathcal{B}) = \int_0^1 d_H({}^\alpha A, {}^\alpha B) d\alpha. \tag{7}$$

A main drawback with this approach is that $d_H^\alpha(\mathcal{A}, \mathcal{B}) = \infty$ if the heights of the two observed fuzzy sets are not the same. Several variations have been proposed to solve this problem [4, 6, 15, 10]. However, a perfect solution for this problem is not found yet [5].

Even if most of the distances between crisp sets of points rely on the point-to-set distance, the distances between two fuzzy sets are usually not defined using the point-to-set distance for fuzzy sets. Two definitions of the point-to-set distance for fuzzy sets are proposed in [1]. The first definition is based on integration over α -cuts, where the distance between point a and fuzzy set \mathcal{B} is defined as

$$d(a, \mathcal{B}) = \int_0^1 d(a, {}^\alpha B) d\alpha = \int_0^1 \min_{b \in {}^\alpha B} d(a, b) d\alpha. \tag{8}$$

The second definition is based on weighting of the points from the support, $Supp(\mathcal{B})$, of the fuzzy set \mathcal{B} with their membership values,

$$d(a, \mathcal{B}) = \min_{b \in Supp(\mathcal{B})} (d(a, b) \cdot F(\mu_B(b))), \tag{9}$$

where $F(t)$ is a decreasing function of t . The point-to-point distance $d(a, b)$ is the spatial distance between two points and is independent on their membership values.

A point-to-set distance can also be defined using fuzzy morphology, but a value of such distance is in general represented by a fuzzy number [1].

3 The Sum of Minimal Distances and Complement Weighted Sum of Minimal Distances for Fuzzy Sets

In this section we extend the Sum of minimal distances and the Complement weighted sum of minimal distances to distances between fuzzy sets. For that purpose we use (6) and (8). A requirement imposed by both (6) and (8) is that observed fuzzy sets have the same height, since, otherwise, integration over α -cuts leads to that the distance is equal to infinity (similar as for the Hausdorff distance on fuzzy sets).

Sum of minimal distances and Complement weighted sum of minimal distances for fuzzy sets can be defined using (6) as

$$d_{SMD}^\alpha(\mathcal{A}, \mathcal{B}) = \int_0^1 d_{SMD}(\alpha A, \alpha B) d\alpha, \tag{10}$$

$$d_{CW}^\alpha(\mathcal{A}, \mathcal{B}) = \int_0^1 d_{CW}(\alpha A, \alpha B) d\alpha. \tag{11}$$

Instead, using (8), Sum of minimal distances and Complement weighted sum of minimal distances for fuzzy sets are defined as

$$d_{SMD}^{ps}(\mathcal{A}, \mathcal{B}) = \frac{1}{2} \left(\sum_{a \in Supp(\mathcal{A})} d(a, \mathcal{B}) + \sum_{b \in Supp(\mathcal{B})} d(b, \mathcal{A}) \right), \tag{12}$$

$$d_{CW}^{ps}(\mathcal{A}, \mathcal{B}) = \frac{1}{2} \left(\frac{\sum_{a \in Supp(\mathcal{A})} d(a, \mathcal{B}) \cdot d(a, \overline{\mathcal{A}})}{\sum_{a \in Supp(\mathcal{A})} d(a, \overline{\mathcal{A}})} + \frac{\sum_{b \in Supp(\mathcal{B})} d(b, \mathcal{A}) \cdot d(b, \overline{\mathcal{B}})}{\sum_{b \in Supp(\mathcal{B})} d(b, \overline{\mathcal{B}})} \right). \tag{13}$$

Similarly, the Hausdorff distance using (8) has the form

$$d_H^{ps}(\mathcal{A}, \mathcal{B}) = \max \left(\sup_{a \in Supp(\mathcal{A})} d(a, \mathcal{B}), \sup_{b \in Supp(\mathcal{B})} d(b, \mathcal{A}) \right). \tag{14}$$

We do not use point-to-set distance (9) since it is not clear which function F to use. We performed test with different decreasing functions, but we did not observe any good performance. Furthermore, if $a \in Supp(\mathcal{B})$, then for $b = a$

$$d(a, b) = 0 \Rightarrow d(a, b) \cdot F(\mu(b)) = 0 \Rightarrow d(a, \mathcal{B}) = 0,$$

which, we feel, further reduces the discriminative power, and thereby the usefulness of this distance definition.

4 Evaluation

This section presents the results of an empirical study of the observed distances. The distances are studied with respect to monotonicity, as well as with respect to noise sensitivity. In addition we compare classification performance of the best of the proposed distances for fuzzy sets, with the best performing one for crisp sets, on fuzzy and crisp discrete representations of objects, respectively.

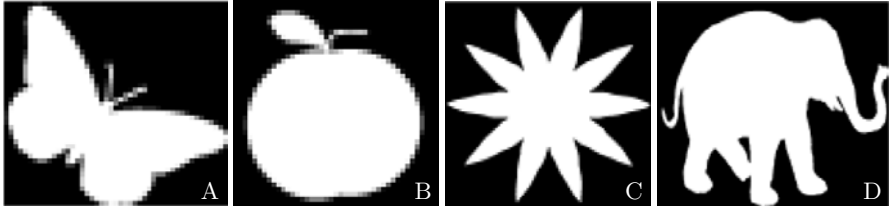


Fig. 2. Examples of tested fuzzy objects

4.1 Evaluation of Monotonicity in Noise Free and Noisy Conditions

For a good matching or registration performance it is desirable that the distance monotonically increases with increasing translation and rotation of the object. Therefore, we study how the distance changes when an object is translated and rotated with respect to the non-transformed object. We say that *monotonicity of a distance measure*, w.r.t. translation (rotation) is fulfilled for a particular object, if the distance between the object and a translated (rotated) version of the same object does not decrease with increasing translation (rotation).

Translation is performed in steps of one pixel horizontally, up to the width of the considered object. Rotation is performed in positive direction around the center of mass of the object, in steps of one degree, up to 23, and up 45 degrees, in two separate tests. At each step the distance is computed between the transformed and the non-transformed image. This is similar to tests performed in [12] for crisp set distances. Rotation of the discrete objects requires interpolation. We tested linear and nearest neighbor interpolation and we selected nearest neighbor interpolation since it provided better performance.

The tests are performed on fuzzy objects obtained from 100 binary images taken from [13]. To obtain fuzzy objects we perform pixel coverage digitization using sub-sampling by a factor 6, which provides 36 different membership levels. Some of the observed fuzzy objects can be seen in Fig. 2. Since some of the objects are rotationally symmetric (see Fig. 2C), monotonicity of the distance with respect to rotation of such objects can not be expected.

Since the membership functions of all the observed objects are of the same height (equal to one), the requirement for set distances based on integration over α -cuts is satisfied.

For any real application it is important that the distance measure is not too sensitive to noise present in the images. We perform tests with two types of noise perturbing the membership values of the observed objects:

1. Additive noise – pixel values are perturbed by Gaussian noise with zero mean and 0.01 variance.
2. Multiplicative noise – pixel values are perturbed by multiplicative uniformly distributed noise with zero mean and variance 0.04.

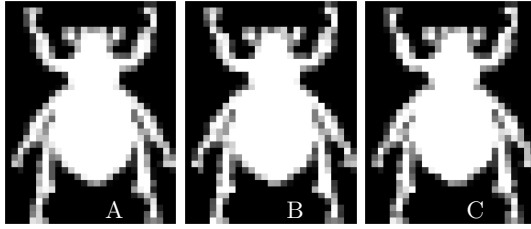


Fig. 3. Different noise conditions. A: Noise free image, B: Image with Additive noise, C: Image with Multiplicative noise

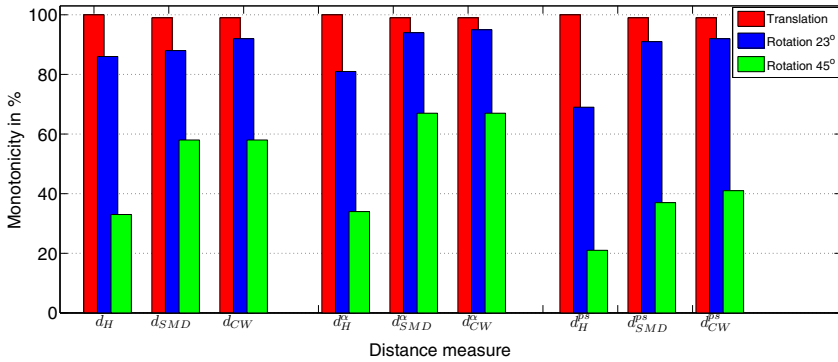


Fig. 4. Percentage of images where monotonicity is fulfilled in noise free case, for each distance and each of the observed rigid transformations

We assume that the pixels with membership value 1 are correctly classified as inner pixels of the object and we, therefore, only apply noise on the boundary of the fuzzy objects. Fig. 3 illustrates the different types of noise.

The proposed distances for fuzzy sets are computed for the fuzzy objects, while corresponding crisp set distances are computed for crisp objects obtained from the considered fuzzy objects using α -cuts at $\alpha = 0.5$. In Fig. 4 we present, for each of the observed distance measures, the percentage of images from the test set for which the distance measure shows monotonic behavior with increasing translation and rotation. In Fig. 5-6 we present, similarly, percentage of images where distances show monotonicity w.r.t. translation and rotation, for the two observed types of noise, and each of the observed distance measures.

We notice that for the noise free case, all the observed distances perform well for the considered test with respect to translation. For the performed rotation tests, the proposed distance measures d_{CW}^α and d_{SMD}^α perform better, while distances d_{CW}^{ps} and d_{SMD}^{ps} perform worse than the corresponding crisp sets distances. The Hausdorff style distances, in general, perform significantly worse than the distances based on all the points of the sets. We conclude that d_{CW}^α exhibits the best overall performance.

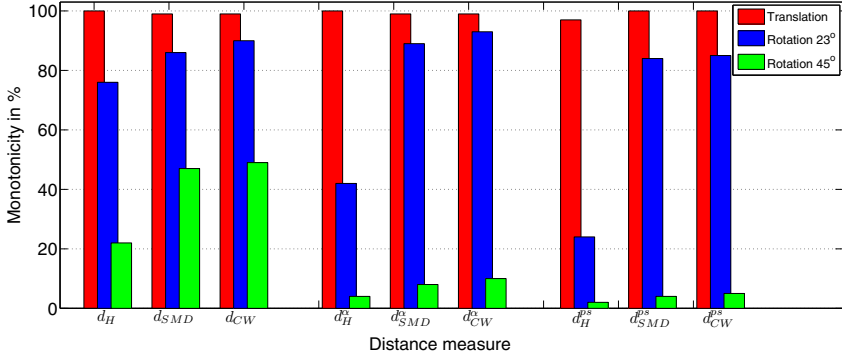


Fig. 5. Percentage of images where monotonicity is fulfilled when Additive noise is applied, for each distance and each of the observed rigid transformations

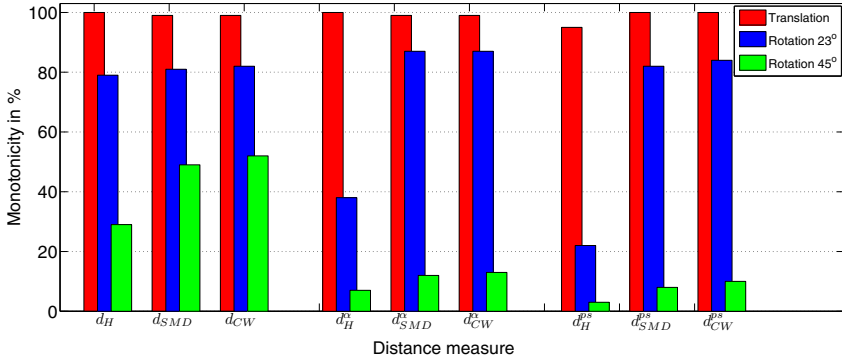


Fig. 6. Percentage of images where monotonicity is fulfilled when Multiplicative noise is applied, for each distance and each of the observed rigid transformations

For both Additive and Multiplicative noise, we observe that the noise has a strong negative effect on monotonicity w.r.t. larger rotations, see Fig. 5-6. Also, we observe that the crisp set distances have significantly better performance than the fuzzy sets distances in the test on monotonicity with respect to rotation up to 45 degrees. Based on these observations, we form the hypothesis that using a reduced number of membership levels can improve performance of the observed distances in noisy conditions. Therefore, we perform tests for monotonicity of the observed distances where the membership levels in the noisy images are quantized to n non-zero levels, where n takes a number of values between 1 (binary case) and 36 (the original number of levels given by the subsampling). Monotonicity of the proposed distances with respect to different number of quantization levels, for both observed types of noise, is presented in Fig. 7-8. We consider only distances defined by (6), since they have the best performance for the noise

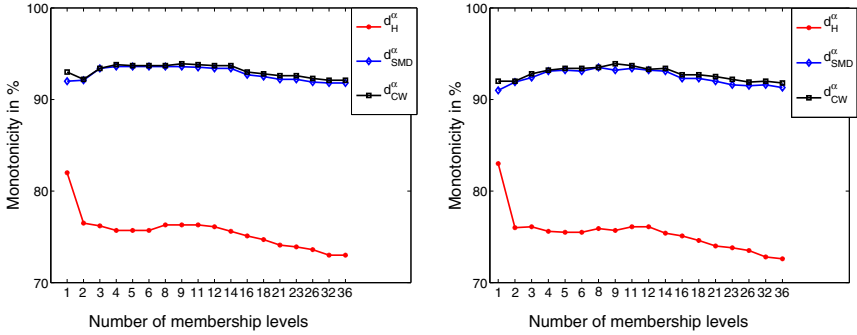


Fig. 7. Percentage of images where distance is non-decreasing with increasing rotation up to 23 degrees, with respect to different number of membership levels for Additive (left) and Multiplicative (right) noise

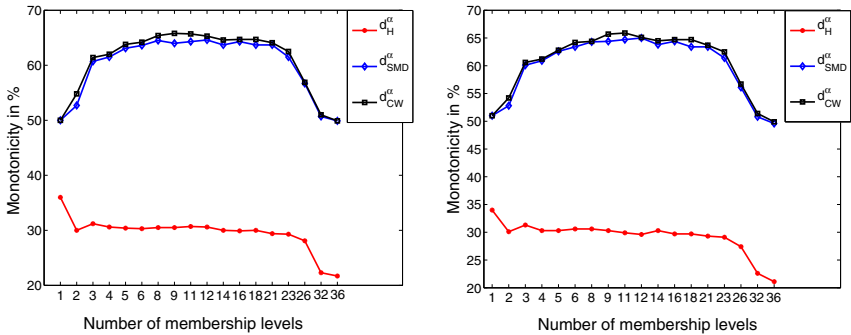


Fig. 8. Percentage of images where distance is non-decreasing with increasing rotation up to 45 degrees, with respect to different number of membership levels for Additive (left) and Multiplicative (right) noise

free case. Using different numbers of quantizations levels has essentially no influence on the monotonicity with respect to translation and we do not present this result graphically. We conclude that d_{CW}^α and d_{SMD}^α , for the given conditions, perform best for membership values quantized to approximately 9 levels. For that case, the distances clearly outperform the corresponding crisp cases (only one non-zero membership level), as can be seen in Fig. 9. Interestingly, d_H^α does not follow the same pattern; monotonicity of d_H^α decreases drastically if more than one membership level is used, and the crisp version performs clearly best.

Fig. 10 shows distance as a function of increasing rotation for one test object with Additive noise, when non-quantized membership levels and memberships quantized to 9 levels, respectively, are used. For this example monotonicity w.r.t. rotation up to 45 degrees is not fulfilled for the non-quantized case, whereas the quantized case does provide monotonicity.

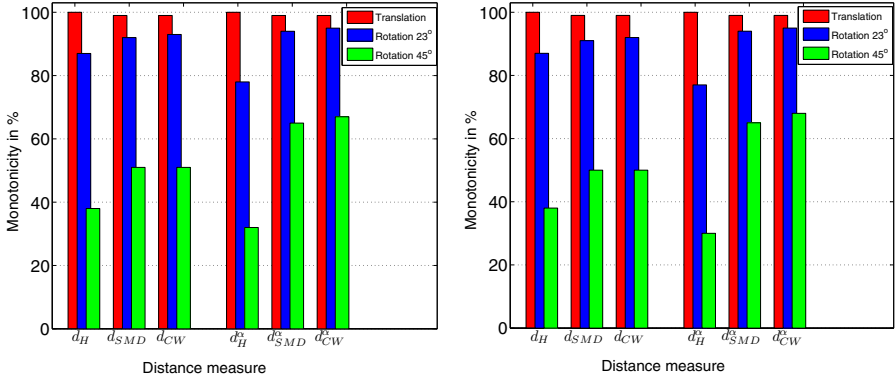


Fig. 9. Percentage of images where monotonicity is fulfilled when Additive (left) and Multiplicative (right) noise is applied and membership values are quantized to one (crisp case) and nine levels, respectively

Observing that the size of the quantization step for the 9-level case is roughly of the same size as the standard deviation of the tested noise, we form the hypothesis that the optimal quantization should essentially hide most of the noise but not more than that. In other words, keeping more membership levels gives the distance measure more information, but only reasonably reliable information leads to corresponding improved performance. Noticing that the membership quantization has a large impact on the performance (compare Fig. 5–6 with Fig. 9), we feel that this issue deserves further studies, and therefore is placed high on our list of future work.

4.2 Comparative Evaluation on Matching Crisp and Fuzzy Objects

In this section we evaluate the performance of the observed distances for object matching. We compare, for a given spatial resolution, classification performance based on fuzzy discrete object representations with classification performance based on crisp discrete object representations. Fuzzy representations of disks and octagons are generated using pixel coverage digitization of continuous crisp disks and octagons, respectively. The corresponding crisp object representations are obtained by taking the α -cut at $\alpha = 0.5$, of the fuzzy object representations (see Fig. 11).

The same procedure is performed for: (i) crisp object representations and (ii) fuzzy object representations, for each studied object size: Five discrete representations of each object type (disk and octagon) are generated as template objects (discretized at random position in \mathbb{Z}^2). 1000 discrete representations of each object type, observations, are generated in the same way. Each observation is then classified as either disk or octagon depending on to which template object an aligned version of the observation has the smallest set distance. In this test

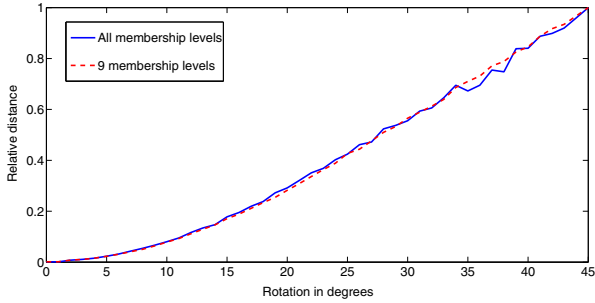


Fig. 10. Rotation of Butterfly image (Fig. 2A) with Additive noise, for different quantization levels. Distance measure for this test is d_{CW}^α (distance values are scaled to the range $[0, 1]$). The quantized version (dashed line) exhibits monotonic behavior for this image, whereas the non-quantized (solid line) does not.

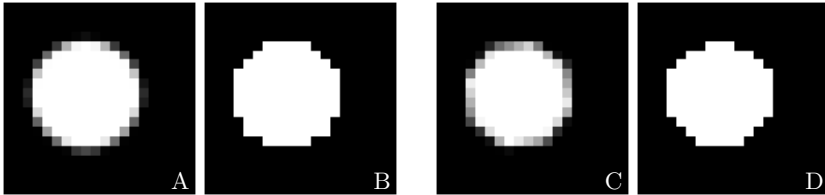


Fig. 11. Classification example. A: Fuzzy discrete representation of a disk, B: Crisp discrete representation of a disk, C: Fuzzy discrete representation of an octagon, D: Crisp discrete representation of an octagon. Objects A, C, and D are correctly classified, while the crisp representation of a disk, B, is incorrectly classified as an octagon.

we use set distances d_{CW} and d_{CW}^α , since they show the best performances in the evaluation of monotonicity (see also [7]). The alignment is performed using greedy search, where monotonicity of the distance measure is essential for success of the procedure. An object is correctly classified if the template object at minimal distance is of the same type (disk or octagon) as the observation.

The number of correctly classified objects, for crisp and fuzzy discrete object representations of a number of sizes, is presented in Fig. 12. We see that the combination of a fuzzy object representation with the proposed distance between fuzzy sets, provides significantly better object discrimination; a higher correct classification ratio, reaching more than 10% of improvement for objects smaller than 10 pixels in diameter, is achieved when using a fuzzy approach than when using the corresponding crisp representation and set distance, at the same spatial resolution. In Fig. 11 an example is shown where, in the presented matching process, fuzzy representations of a disk and an octagon (Fig. 11A and Fig. 11C), as well as a crisp representation of an octagon (Fig. 11D), are correctly classified, whereas a crisp representation of a disk (Fig. 11B) is incorrectly classified as an octagon.

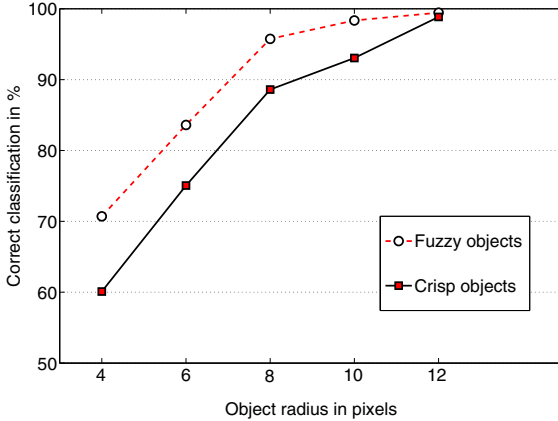


Fig. 12. Correct classification ratios for distance based object classification utilizing fuzzy and crisp discrete object representations, for a range of spatial resolutions

5 Summary and Conclusions

We have presented two different extensions of Sum of minimal distances and Complement weighted sum of minimal distances to fuzzy sets and we have performed an empirical evaluation of the monotonicity of the distance measures with respect to translation and rotation of discrete objects, as well as evaluation of noise sensitivity. The proposed distances, d_{CW}^α and d_{SMD}^α perform better, while distances d_{CW}^{ps} and d_{SMD}^{ps} perform worse than corresponding crisp sets distances. We conclude that the proposed distance d_{CW}^α has the best overall performance.

Based on the observed performance for noisy conditions, we hypothesize that additional quantization of the membership levels may actually lead to improved performance for noisy data. Therefore, we have performed test on noise sensitivity using a reduced number of membership levels. Tests showed that the performance of the observed distance measures depend on the used number of membership levels and that it seems that a quantization step roughly of the same size as the standard deviation of the noise gives best performance. As part of future work, we intend to explore the relationship between the noise level and the appropriate number of quantization levels used in a fuzzy object representation.

We have performed distance based object classification for crisp and fuzzy discrete object representations. We conclude that the combined utilization of a fuzzy object representation and the proposed distance measure, d_{CW}^α , leads to significantly improved performance compared to a corresponding classification based on a crisp object representation. This demonstrates that the proposed distances are capable of utilizing the additional information that a fuzzy object representation provides and that this information can provide improved performance for different related applications in image processing.

Acknowledgments

J. Lindblad and N. Sladoje are financially supported by the Ministry of Science of the Republic of Serbia through the Projects ON174008 and III44006 of the Mathematical Institute of the Serbian Academy of Science and Arts. Scientific support from Prof. Gunilla Borgefors is gratefully acknowledged.

References

1. Bloch, I., Maitre, H.: Fuzzy distances and image processing, In: Proc. of the ACM Symposium on Applied Computing, Tennessee, USA, pp. 570–573 (1995)
2. Bloch, I.: On fuzzy distances and their use in image processing under imprecision. *Pattern Recognition Letters* 32, 1873–1895 (1999)
3. Borgefors, G.: Hierarchical Chamfer Matching: A Parametric Edge Matching Algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 10(6), 849–865 (1988)
4. Boxer, L.: On Hausdorff-like metrics for fuzzy sets. *Pattern Recognition Letters* 18, 115–118 (1997)
5. Brass, P.: On the nonexistence of Hausdorff-like metrics for fuzzy sets. *Pattern Recognition Letters* 23, 39–43 (2002)
6. Chaudhuri, B.B., Rosenfeld, A.: On a metric distance between fuzzy sets. *Pattern Recognition Letters* 17, 1157–1160 (1996)
7. Ćurić, V., Lindblad, J., Sladoje, N., Sarve, H.: Set distances and their performance in binary image registration (2010) (submitted for publication)
8. Dubuisson, M., Jain, A.: A Modified Hausdorff Distance for Object Matching. In: Proc. of International Conference on Pattern Recognition, Jerusalem, Israel, pp. 566–568 (1994)
9. Eiter, T., Mannila, H.: Distance measures for point sets and their computation. *Acta Informatica* 34, 103–133 (1997)
10. Fan, J.: Note on Hausdorff-like metrics for fuzzy sets. *Pattern Recognition Letters* 19, 739–796 (1998)
11. Fudos, I., Palios, L., Pitoura, E.: Geometric-Similarity Retrieval in Large Image Bases. In: International Conference on Data Engineering, pp. 441–450. IEEE, Los Alamitos (2002)
12. Lindblad, J., Ćurić, V., Sladoje, N.: On set distances and their application in image analysis. In: Proc. of International Symposium on Image and Signal Processing and Analysis (ISPA), pp. 449–455. IEEE, Salzburg (2009)
13. MPEG7 CE Shape-1 Part B, URL(2010), <http://www.imageprocessingplace.com>
14. Ralescu, A., Ralescu, D.: Probability and fuzziness. *Information Sciences* 34(2), 85–92 (1984)
15. Rosenfeld, A.: Distances between fuzzy sets. *Pattern Recognition Letters* 3, 229–233 (1985)
16. Sladoje, N., Lindblad, J.: High-precision boundary length estimation by utilizing gray-level information. *IEEE Transaction on Pattern Analysis and Machine Intelligence* 31(2), 357–363 (2009)
17. Sladoje, N., Lindblad, J.: Estimation of moments of digitized objects with fuzzy borders. In: Roli, F., Vitulano, S. (eds.) ICIAP 2005. LNCS, vol. 3617, pp. 188–195. Springer, Heidelberg (2005)
18. Sladoje, N., Nyström, I., Saha, K.P.: Measurements of digitized objects with fuzzy borders in 2D and 3D. *Image Vision Computing* 23(2), 123–132 (2005)
19. Zadeh, L.: Fuzzy sets. *Information and control* 8, 338–353 (1965)

Unsupervised Polygonal Reconstruction of Noisy Contours by a Discrete Irregular Approach

Antoine Vacavant^{1,*}, Tristan Roussillon^{2,3}, and Bertrand Kerautret^{4,5}

¹ Clermont Université, Université d’Auvergne, ISIT, 63000, France
`antoine.vacavant@iut.u-clermont1.fr`

² Université de Lyon, CNRS

³ Université Lyon 2, LIRIS, UMR5205, 69676, France
`tristan.roussillon@liris.cnrs.fr`

⁴ LORIA, UMR 7503 CNRS, Université de Nancy, France

⁵ LAMA, UMR 5127 CNRS, Université de Savoie, 73376, France
`kerautre@loria.fr`

Abstract. In this paper, we present an original algorithm to build a polygonal reconstruction of noisy digital contours. For this purpose, we first improve an algorithm devoted to the vectorization of discrete irregular isothetic objects. Afterwards we propose to use it to define a reconstruction process of noisy digital contours. More precisely, we use a local noise detector, introduced by Kerautret and Lachaud in IW-CIA 2009, that builds a multi-scale representation of the digital contour, which is composed of pixels of various size depending of the local amount of noise. Finally, we compare our approach with previous works, by considering the Hausdorff distance and the error on tangent orientations of the computed line segments to the original perfect contour. Thanks to both synthetic and real noisy objects, we show that our approach has interesting performance, and could be integrated into document analysis systems.

1 Introduction

The representation of graphical objects (such as symbols, line drawings, characters, *etc.*) with line segments is an important task for various document and image analysis applications. This vectorization stage has been widely studied since the 90’s, and many algorithms have been designed [4, 6, 18]. Discrete or digital contours are natural outputs of image segmentation algorithms or digitization processes (*e.g.* document scanning). In most cases, digital contours are not perfect digitizations of ideal shapes but present noise and irregularities. In this case, classical approaches of contour detection generally need a parameter, and the output has to be filtered and post-processed (see Fig. 1 for an example with the Canny edge detector).

* This work has been supported by the French National Agency for Research with the reference ANR-10-CORD-005 (REVES project).

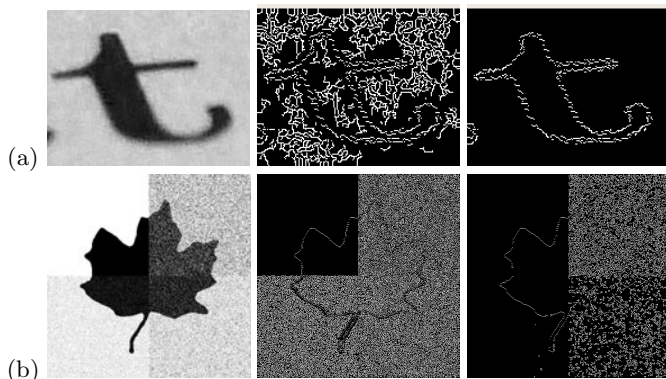


Fig. 1. The Canny edge detector applied on two images with two sets of parameters. For image (a), even if we could obtain an interesting result, a post-process is necessary to filter the output of the detector in order to compute a linear contour. A very noisy image (b) cannot be efficiently handled by this detector, even with various parameters.

Lately, two different approaches have been proposed in the digital geometry community. (i) The noisy digital contour (or a thick digital curve around it) is partitioned into thick (or blurred) segments [13,5]. This last approach requires a global thickness parameter and thus cannot handle noises that are not uniform. (ii) To cope with this problem an adaptive pixel resizing method has been proposed in [12]. The idea is interesting but its implementation (as described in [12]) has several drawbacks. Firstly, the resizing function (which is not explicitly given) is based on the length of the symmetric tangents computed on the digital contour at the initial scale. Second, the resized pixels overlap so that the polygonalization is performed by a complex generalized preimage algorithm. Finally, the set of resized pixels may not be homotopic to the input digital contour and the topological control process proposed by the authors requires a skeleton computation.

In this paper, we propose a novel approach to compute a polygonal reconstruction from a noisy digital contour. For this purpose, we compute a set of resized pixels from a noisy digital contour thanks to a local noise detector [7]. The idea is to locally look at the length of the maximal digital straight segments lying on the input digital contour at decreasing resolutions. The resolution beyond which we observe that the evolution of the length of the maximal segments is similar to the theoretical behavior for digitizations of smooth shapes does not contain any noise. The pixels at this resolution corresponds to bigger pixels at the initial resolution so that the higher the amount of noise is, the biggest the pixels are.

This set of resized pixels is transformed into an irregular isothetic object composed of rectangular cells and whose topology is stored into a Reeb graph [15]. For the reconstruction, each arc is vectorized into a polygonal line. The polygonal lines are then linked together so that the resulting polygonal structure reflects the topology of the irregular isothetic object.

In [15], arcs are vectorized following a visibility cone approach. Even if this reconstruction method has a linear-time complexity, it is a greedy approach that may induce an increasing error that leads to some very short segments and very acute turn angles. In this paper, we segment each arc into straight parts in linear-time using O'Rourke's algorithm [10], which is much simpler than the generalized preimage approach of [12]. We then propose two different reconstructions: (i) the first one takes into account the shape of the preimage of each straight part so that the resulting polygon lies within the irregular isothetic object, (ii) the second one is a simpler method, more convenient for objects that contain straight parts, but the resulting polygon may not completely lie within the irregular isothetic object.

In the next section, we recall definitions about irregular isothetic objects, and the previous work of [15]. We then describe our polygonalization methods and use it in order to reconstruct a noisy digital contour. Finally, we present several experiments and comparisons.

2 Preamble and Previous Work

2.1 Definitions

In this section, we first recall the concept of irregular isothetic grids (\mathbb{I} -grids) in 2-D, with the following definition [3][17].

Definition 1 (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 non overlapping rectangular cells whose edges are parallel to the X and Y axes. 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 $(l_R^x, l_R^y) \in \mathbb{R}_+^{*2}$.*

We say that two cells R_1 and R_2 are *ve*-adjacent if they share either a vertex or an edge, and *e*-adjacent if they share an edge. In a more general way, we say that R_1 and R_2 are *k*-adjacent, and *k* may be interpreted as *e* or *ve* in the following definitions. A set of cells \mathcal{E} is a *k*-arc iff for each element of $\mathcal{E} = \{R_i\}_{1 \leq i \leq n}$, R_i has exactly two *k*-adjacent cells, except R_1 and R_n . A set of cells \mathcal{E} is a *k*-object iff for each couple of cells $(R_1, R_2) \in \mathcal{E} \times \mathcal{E}$, there exists a *k*-arc between R_1 and R_2 in \mathcal{E} (Fig. 2 left).

We consider an order relation based on the cells 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 $x_R - (l_R^x/2)$ and for sake of clarity we write it $R^L = x_R - (l_R^x/2)$.

Definition 2 (Order relation 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 relation \preceq^L , based on the cells borders:*

$$\forall R_1, R_2 \in G, R_1 \preceq^L R_2 \Leftrightarrow R_1^L < R_2^L \vee (R_1^L = R_2^L \wedge R_1^T \leq R_2^T).$$

This order relation is of great importance either for the Reeb graph computation or for the segmentation of each *k*-arc into straight parts using O'Rourke algorithm, which requires that the input ranges have increasing x-coordinate.

2.2 Previous Algorithm for Irregular Object Vectorization

The work of Vacavant *et al.* [15] aimed to develop an algorithm that vectorizes a k -object with line segments. This method is divided into two main steps.

Representation of the Topology of an Irregular Object. The Reeb graph [11] of the input irregular k -object \mathcal{E} is a way of representing the topology of \mathcal{E} . The k -object \mathcal{E} is scanned from left to right according to the order induced by \preceq^L , given in Definition 2 (see Fig. 2 for an example).

At the beginning, the intersection between \mathcal{E} and the scanning vertical line has only one connected part and the Reeb graph is created with one edge between two nodes (b for begin and e for end). If a connected part splits into several parts, we add a node (s for split) from which start as many edges as there are parts. Conversely, if two connected parts merge, we link the corresponding edges to a node (m for merge) (see Fig. 2).

Moreover, the initial set of cells is recoded into a new one (without changing the shape of the object however) so that each edge of the Reeb graph corresponds to a k -arc having cells of increasing left border. This is done during the scan. We merge with the cell having the smallest left border, all its k -adjacent cells by using the following update procedure.

Update procedure. Let A be a k -arc, and R_1 and R_2 two adjacent cells of \mathcal{E} such that $R_1 \in A$, $R_1^L < R_2^L$, and R_2 adjacent to A (and thus should be added to A). If $R_2^L = R_1^R$, one just add R_2 to A , else the procedure *updates* the k -arc A with R_2 , and may recode A . For that, it first builds the *greatest common rectangle* (GCR) F_2 of R_1 and R_2 . This GCR is the greatest rectangle that can be contained in $R_1 \cup R_2$ [15]. Then, Vacavant *et al.* consider the rectangles $R_1 - F_2$ and $R_2 - F_2$. If $R_1^R < R_2^R$, they denote $R_1 - F_2 = F_1$ and $R_2 - F_2 = F_3$. The k -arc A is finally updated with respect to five main configurations (see Fig. 3).

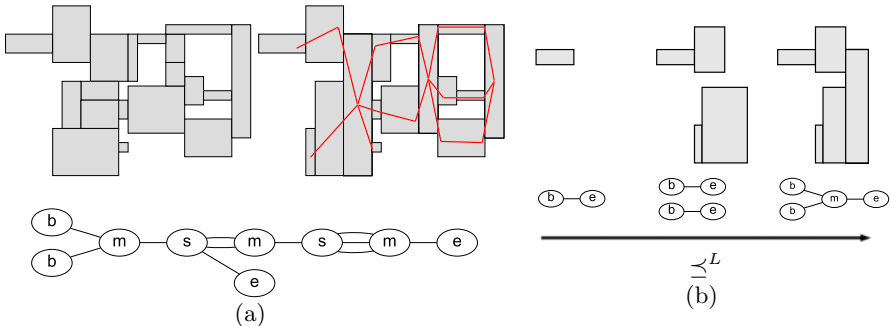


Fig. 2. (a) An example of an irregular object \mathcal{E} (left), the final recoded structure with arcs, the obtained polygonalization (right) and the Reeb graph associated to the order defined on \mathcal{E} (bottom) [16]. In (b), we show the recognized k -arcs and the associated Reeb graph for some iterations of this algorithm, in respect to the \preceq^L order.

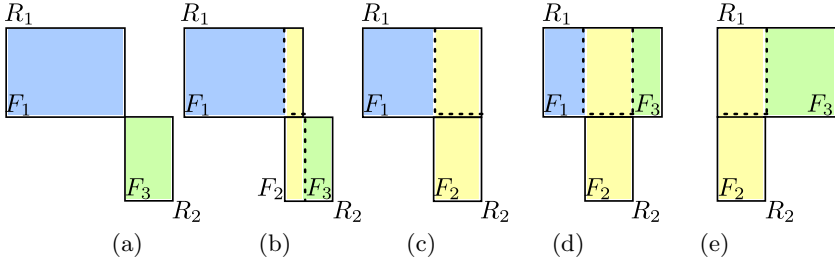


Fig. 3. Description of rectangles F_1 , F_2 and F_3 in the update procedure. When $R_1^R < R_2^R$ (a and b), $R_1 - F_2 = F_1$ and $R_2 - F_2 = F_3$, else $R_1 - F_2 = \{F_1, F_3\}$ (c, d and e). If $R_1^R = R_2^L$, $F_2 = \emptyset$, when $R_1^R = R_2^R$, $F_3 = \emptyset$ and finally $F_1 = \emptyset$ in the case $R_1^L = R_2^L$.

Polygonal Reconstruction of an Irregular Object. The construction of the polygonal structure of \mathcal{E} is performed by reconstructing each k -arc that recodes \mathcal{E} . This stage is driven thanks to a visibility cone approach inspired from [14] (see also Fig. 4). Even if this algorithm is linear-time, it is a greedy approach that could leads to some very short segments and acute angles.

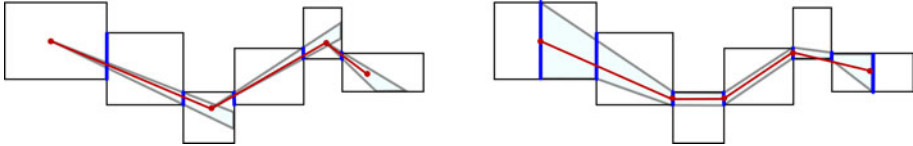


Fig. 4. The visibility cone approach incrementally produces a polygonal reconstruction with a partial preimage (left). In our contribution, we use the O’Rourke’s algorithm in order to obtain a complete preimage representation (right). We also show an example of reconstruction that we describe in the section 3.

3 Unsupervised Polygonalization of Noisy Digital Contours

3.1 A Novel Approach to Vectorize Irregular Isothetic Objects

The polygonal reconstruction of \mathcal{E} is again performed by reconstructing each k -arc that recodes \mathcal{E} . Though, instead of decomposing a given k -arc A into segments lying into a visibility cone like in [15], we decompose it into *straight k -arcs*, *i.e.* sets of k -adjacent cells that can cover a straight line and whose preimage is thus not empty. In Fig. 4 (right), we present the result of this kind of process on a simple k -arc decomposed into four straight parts.

Originally, O’Rourke’s algorithm [10] aimed to solve a linear inequality system. Given n ranges $\{[\alpha_k, \omega_k]\}_{k=1,n}$ ordered by time t_k , with $t_1 < \dots < t_n$, this approach computes all the lines $u = mt + b$ that pass through each range,

i.e. all pairs (m, b) such that $\alpha_k \leq mt_k + b \leq \omega_k$. In our case, the input ranges are the intersections between two successive cells of A . These intersections are vertical straight segments (possibly degenerated as a point) whose the extremity of greatest (resp. smallest) y-coordinate is called upper (resp. lower) input point. Due to the construction of the k -arc, the vertical straight segments are of increasing x-coordinate and O'Rourke's algorithm can thus be applied in order to compute the preimage of each straight part of A .

Even if O'Rourke originally explains its algorithm in the dual plane (m, b) [10], we can avoid explicit transformations and only work in the primal plane (u, t) . The preimage is implicitly described by some consecutive vertices of the lower (denoted by \mathcal{L}) and upper (denoted by \mathcal{U}) convex hull of respectively the upper and lower input points.

We now describe a first algorithm that takes into account the shape of the preimage of each straight parts (Algorithm 1-C2, lines 7-25). It computes a polygonal line that completely lies within the k -arc.

In Fig. 5, we depict several iterations of this algorithm on a straight part. Points p_a and p_c are initialized as the first two points of \mathcal{U} , while p_b and p_d as the ones of \mathcal{L} (Fig. 5(a)). If $p_c.x > p_d.x$ (i), we move forward p_b and p_d , whereas if $p_c.x < p_d.x$ (ii), we move forward p_a and p_c . If $p_c.x = p_d.x$ (ii), we move both pairs of points. In either case, the middle of the intersection between a vertical line passing by $p_c.x$ in case (i) (Fig. 5(c)) or $p_d.x$ in case (ii) (Fig. 5(b)) and the preimage is the new vertex of the polygonal reconstruction. The process is linear-time and the resulting polygonal line lies inside the k -arc.

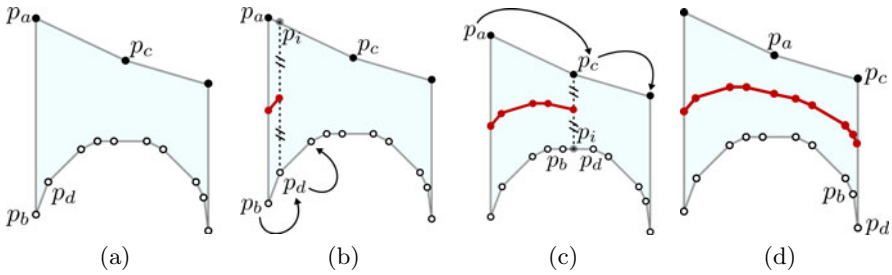


Fig. 5. Illustration of Algorithm 1-C2 on a preimage obtained from O'Rourke process. (a) is the initialization step, (b) is an update step of the reconstruction implying a lower input point, (c) a upper one, and (d) is the obtained polygonal reconstruction of the underlying straight k -arc. Algorithm 1-S2 consists in joining the first and the last point of this reconstruction.

We also develop an other algorithm (Algorithm 1-S2, lines 4-6) that constructs one straight line per k -arc passing through the middle of the first and last input range. Even if the resulting polygonal line may be partly out of the k -object, this is an interesting way of decomposing an irregular object into a few line segments.

We show in Fig. 6 an example of the use of our contribution (computation of the complete preimage and reconstruction into line segments with

Algorithm 1. Polygonal reconstruction of a straight k -arc based on its preimage

input : a preimage \mathcal{P} computed from a straight k -arc A , represented with its upper convex hull points \mathcal{U} of size $n_{\mathcal{U}}$ and the lower convex hull points \mathcal{L} of size $n_{\mathcal{L}}$, and the version selected

output: a polygonal structure computed inside \mathcal{P} , stored in the list of points \mathcal{R}

```

1  $u \leftarrow 0, l \leftarrow 0$ ;
2  $p_a \leftarrow \mathcal{U}[u], p_c \leftarrow \mathcal{U}[u+1], p_b \leftarrow \mathcal{L}[l], p_d \leftarrow \mathcal{L}[l+1]$ ;
3  $d \leftarrow p_c.x - p_d.x$ ;
4 if  $version = S2$  then                                     {Version S2: Simple and Straight reconstruction}
5    $p_1 \leftarrow \text{middle}(\mathcal{L}[0], \mathcal{U}[0])$ ;
6    $p_2 \leftarrow \text{middle}(\mathcal{L}[n_{\mathcal{L}}-1], \mathcal{U}[n_{\mathcal{U}}-1])$ ;
7    $\mathcal{R} \leftarrow \{p_1, p_2\}$ ;
8 if  $version = C2$  then                                     {Version C2: Complete and Curved reconstruction}
9   do
10    while  $d < 0 \wedge u < n_{\mathcal{U}}$  do                             {Update  $\mathcal{R}$  from upper convex hull}
11       $\Delta$ : line of equation  $x = p_c.x, p_l \leftarrow \Delta \cap [p_b, p_d]$ ;
12       $\mathcal{R} \leftarrow \mathcal{R} \cup \text{middle}(p_l, p_c)$ ;
13       $u \leftarrow u + 1$ ;
14       $p_a \leftarrow p_c, p_c \leftarrow \mathcal{U}[u]$ ;
15       $d \leftarrow p_c.x - p_d.x$ ;
16    while  $d > 0 \wedge l < n_{\mathcal{L}}$  do                             {Update  $\mathcal{R}$  from lower convex hull}
17       $\Delta$ : line of equation  $x = p_d.x, p_l \leftarrow \Delta \cap [p_a, p_c]$ ;
18       $\mathcal{R} \leftarrow \mathcal{R} \cup \text{middle}(p_l, p_d)$ ;
19       $l \leftarrow l + 1$ ;
20       $p_b \leftarrow p_d, p_d \leftarrow \mathcal{L}[l]$ ;
21       $d \leftarrow p_c.x - p_d.x$ ;
22    while  $d = 0 \wedge u < n_{\mathcal{U}} \wedge l < n_{\mathcal{L}}$  do {Update  $\mathcal{R}$  from upper and lower convex hulls}
23       $\mathcal{R} \leftarrow \mathcal{R} \cup \text{middle}(p_c, p_d)$ ;
24       $u \leftarrow u + 1, l \leftarrow l + 1$ ;
25       $p_a \leftarrow p_c, p_c \leftarrow \mathcal{U}[u], p_b \leftarrow p_d, p_d \leftarrow \mathcal{L}[l]$ ;
26       $d \leftarrow p_c.x - p_d.x$ ;
27    while  $u < n_{\mathcal{U}} \vee l < n_{\mathcal{L}}$ ;
28 return  $\mathcal{R}$ ;

```

Algorithm 1-C2) on an irregular object, result of a quadtree decomposition. In the following, we show how to use these algorithms to vectorize noisy digital contours.

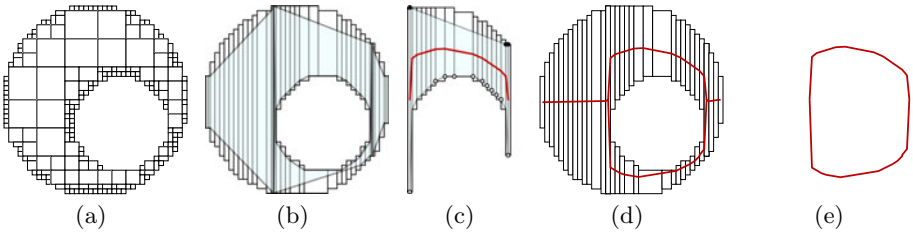


Fig. 6. Illustration of our contribution on an object digitized with a quadtree (a). (b) is the complete preimage computed on each k -arc encoding the object. One could note that the k -arc at the bottom is decomposed into two straight k -arcs. In (c), we present the reconstruction of a single k -arc, and the associated preimage and upper/lower convex hull points. We also depict the complete polygonal reconstruction of the object, constructed inside the preimage (d), and the final contour obtained with our filtering procedure explained in Section 3.2.

3.2 Polygonalization of Noisy Contours by an Irregular Discrete Approach

We now propose to analyze noisy digital contour by using Kerautret and Lachaud’s local noise detector [7]. This a method for estimating locally if the digital contour is damaged, what is the amount of noise and what are the highest resolution at which a part of the contour should be considered as noise-free. Depending on the selected resolution, a part of the contour is covered by a pixel of a given size at the initial resolution. The higher the amount of noise is, the biggest the pixels are.

In Fig. 7(b), we show an example of the output of this parameter-free algorithm applied to the noisy digital object depicted in Fig. 7(a).

As shown in Fig. 7(b), the resized pixels overlap and thus cannot be viewed as an irregular isothetic object (Definition 1). However each resized pixel contain a given number of pixels (at the initial resolution) so that the set of resized pixels cover a subset of the input image. This subset, which is an irregular isothetic object, is the input of our reconstruction method described in the previous sections.

The input digital contour is always homotopic to a ring (one connected component and one hole). However, as in [12], the set of resized pixels may not be homotopic to the input digital contour. We can imagine that the set of resized pixels may not contain any hole or may contain more than one hole.

Thanks to the Reeb graph, which encodes the topology of the input object, we can decide whether we are in a general case (one cycle) or not (none or more than one cycle).

Moreover, in the general case, we can choose to not process the k -arcs that do not belong to the cycle so that the polygonal reconstruction is a simple polygon (Algorithm 2). For instance, only reconstructing the cycle linking nodes s, m, m, m in Fig. 7(d) is a way of avoiding extra polygonal lines pointed by arrows in Fig. 7(c).

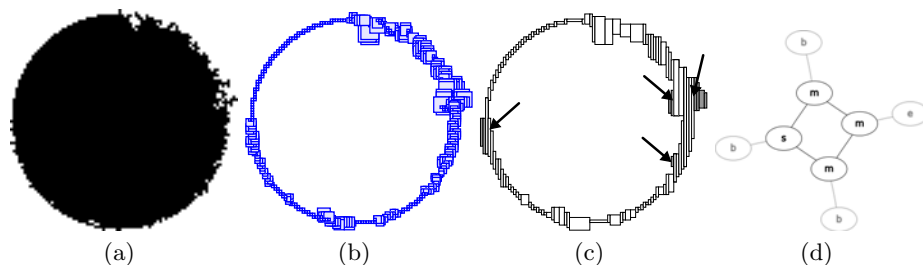


Fig. 7. From a noisy contour (a), we build a set of resized pixels (b). Then, we filter the result of our vectorization algorithm by removing k -arcs that do not belong to the polygonal minimal contour (the ones pointed by arrows). To do so, we remove their associated edges in the Reeb graph (d).

Algorithm 2. Filtering of the k -arcs and the Reeb graph encoding a noisy object

input : the set of k -arcs recoding it \mathcal{A} and its associated Reeb graph \mathcal{G}
output: the sets \mathcal{A}, \mathcal{G} are filtered in order to obtain a polygonal contour

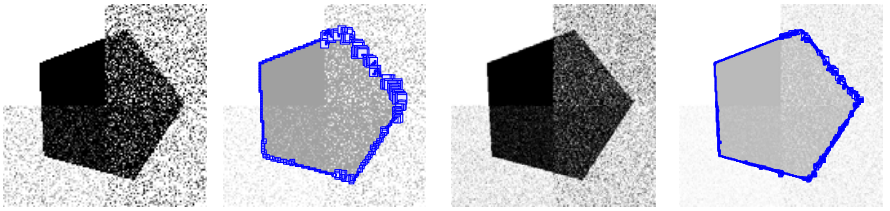
```

1 foreach  $k$ -arc  $a \in \mathcal{A}$  do
2    $x$ : the associated edge of  $a$  in  $\mathcal{G}$  ;
3   if  $x = b - m \vee x = b - s \vee x = s - e \vee x = m - e$  then
4     remove  $x$  from  $\mathcal{G}$  ;
5     remove  $a$  from  $\mathcal{A}$  ;
6 return  $\mathcal{A}, \mathcal{G}$  ;

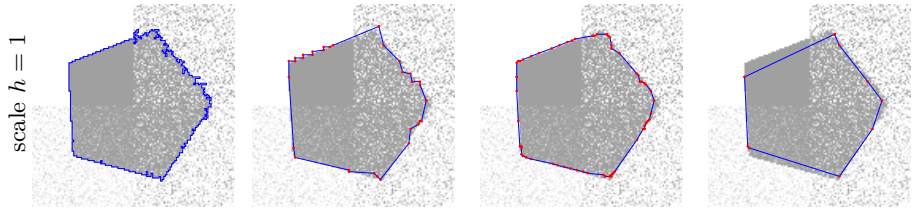
```

4 Experimental Results

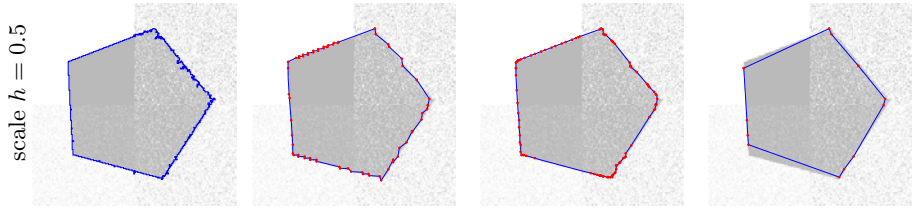
To experiment the quality of the proposed approach, we first consider a polygonal shape that was perturbed by a Gaussian noise, with different standard deviations



(a) source, $h = 1$ (b) multi-scale levels (c) source, $h = 0.5$ (d) multi-scale levels



(e) noisy contour $n = 544$ (f) Alg-VC, $n = 33$ $E_d = 0.80, \theta_{err}^2 = 0.12$ (g) Alg1-C2, $n = 60$ $E_d = 0.83, \theta_{err}^2 = 0.06$ (h) Alg1-S2, $n = 9$ $E_d = 2.73, \theta_{err}^2 = 0.07$



(i) noisy contour $n = 1004$ (j) Alg-VC, $n = 69$ $E_d = 0.74, \theta_{err}^2 = 0.12$ (k) Alg1-C2, $n = 91$ $E_d = 0.65, \theta_{err}^2 = 0.04$ (l) Alg1-S2, $n = 12$ $E_d = 2.99, \theta_{err}^2 = 0.05$

Fig. 8. Illustration of the reconstruction algorithms. The first row shows the multi-scale levels obtained from the source contours (e,i). The second and third rows show error measures for algorithms Alg-VC (that uses previous work), Alg1-C2 and Alg1-S2 described in this article. These results were obtained with resp. the scale $h = 1$ and 0.5 .

($\sigma_0 = 0$, $\sigma_1 = 75$, $\sigma_2 = 125$, $\sigma_3 = 175$). These images were generated with two different grid sizes $h = 1$ and 0.5 (Fig. 8 (a,c)). The resized pixels (illustrated on images of Fig. 8 (b,d)) were obtained from the digital contours extracted by using a simple threshold (set to 128) (images (e,i)). The quality measures were given by the total number of points (n), the mean minimal euclidean distance (E_d) between the source contour points P_i to the resulting polygon, and the error on tangent orientations (θ_{err}^2). The measure E_d was obtained after associating each contour points P_i of the initial shape (non noisy) to the nearest consecutive vertex pair V_k, V_{k+1} . These associations were also used to determine the tangent error θ_{err}^2 where θ_{err} is the angle between the tangent vector defined from V_k, V_{k+1} and the tangent provided by the $\lambda - MST$ estimator [8] applied on the source discrete contour.

The experiments confirm the awaited improvements provided by the Algorithm \square -C2 (Alg1-C2 in short) in comparison with the use of the algorithm based on visibility cone [15] (denoted as Alg-VC). It is visible especially for the tangent error measure θ_{err}^2 but also for the distance error E_d . The second variant Algorithm \square -S2 (Alg1-S2 in short) produces a more compact representation while preserving a moderate tangent error θ_{err}^2 . However this last algorithm is less convenient on the point of view of the E_d error.

The Algorithm \square -C2 was also experimented on real images of characters, acquired from a photographed document. A given threshold was used to extract the digital contours on which the resized pixels were computed (as illustrated on the second row Fig. 9). We thus show that our vectorization algorithm could be applied in document analysis systems.

Finally, we compare our methods with algorithms developed by Nguyen and Rennesson [9] which are based on a global optimization scheme in association

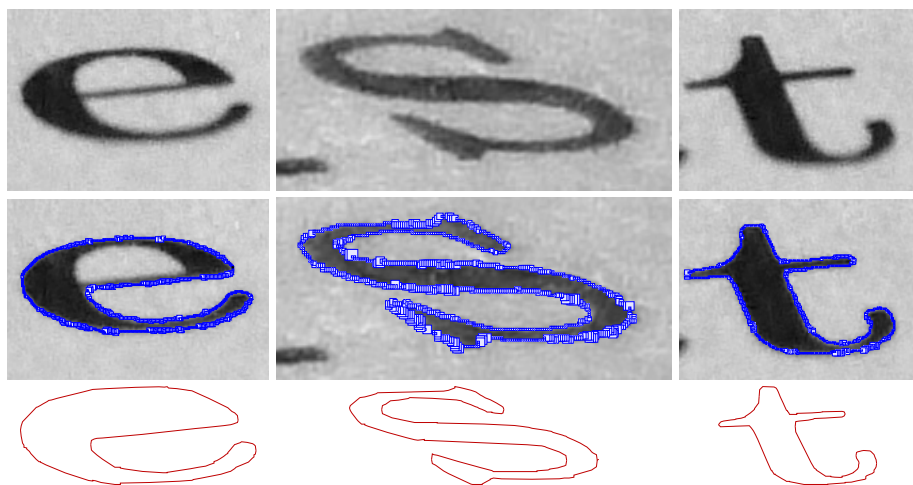


Fig. 9. The meaningful boxes extracted from scanned characters (center), and the final reconstruction we propose with Alg1-C2 (bottom)

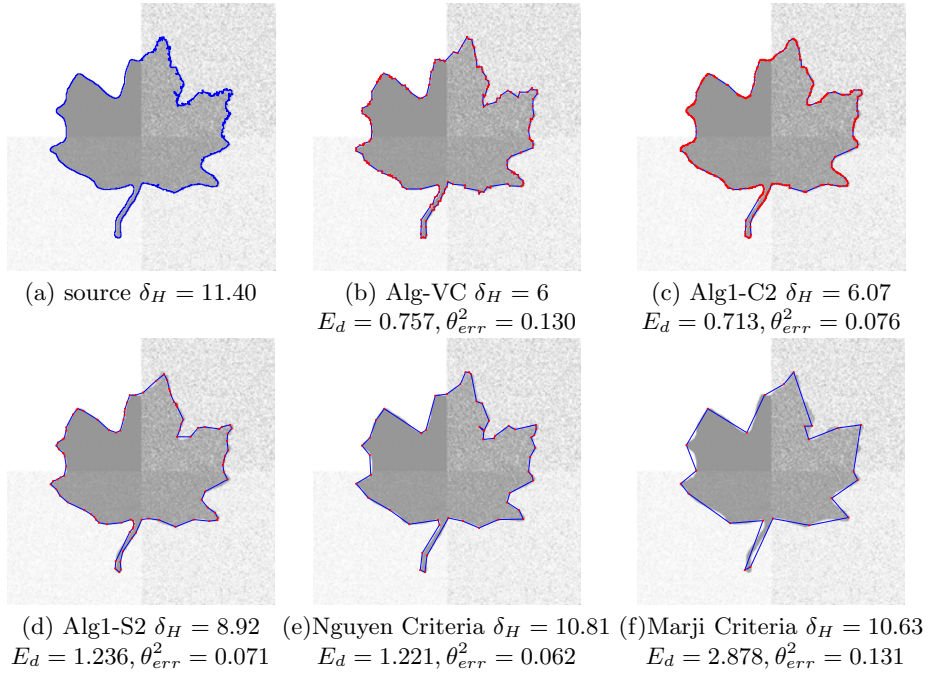


Fig. 10. Comparisons of the proposed approaches with others recent parameter free approaches [9]

with the Marji’s criteria (MC) or another one proposed by the authors (NC). In Fig. 10, we present the polygonal contour obtained from our methods, and from the NC and MC algorithms which are both parameter free approaches. For each experiment, we measure the Hausdorff error (δ_H) and the previously described errors. The comparisons show that the proposed approaches are less compact than both the NC or MC but provide better precision for the δ_H and E_d errors. On the point of view of the tangent orientation error θ_{err}^2 our approaches with Alg1-C2 or Alg1-S2 are comparable with the one of the NC algorithm.

5 Conclusion and Future Works

In this paper, we address the problem of reconstruction of noisy digital contours. We transform the resized pixels obtained by Kerautret and Lachaud’s algorithm [7] into an irregular isothetic object recoded in a set of k -arcs whose topology is stored into a Reeb graph. We then vectorize it with two different linear-time methods that improve a previous work of Vacavant *et al.* [15].

As a future work, we plan to use the Reeb graph to deal with degenerate cases. We also want to consider the possibility to improve the reconstruction by using information of flat and curved parts of the processed noisy objects, since this information can be extracted from the meaningful scale detection [7].

References

1. Canny, J.: A Computational Approach To Edge Detection. *IEEE Trans. on PAMI* 8(6), 679–698 (1986)
2. Coeurjolly, D., Tougne, L.: Digital Straight Line Recognition on Heterogeneous Grids. In: *Proc. of SPIE Vision Geometry XII*, vol. 5300, pp. 108–116 (2004)
3. Coeurjolly, D., Zerarga, L.: Supercover Model, Digital Straight Line Recognition and Curve Reconstruction on the Irregular Isothetic Grids. *C&G* 30(1), 46–53 (2006)
4. Cordella, L.P., Vento, M.: Symbol Recognition in Documents: a Collection of Techniques? *Int. Journal on Document Analysis and Recognition* 3(2), 73–88 (2000)
5. Faure, A., Feschet, F.: Linear Decomposition of Planar Shapes. In: *ICPR*, pp. 1096–1099 (2010)
6. Hilaire, X., Tombre, K.: Robust and Accurate Vectorization of Line Drawings. *IEEE Trans. on PAMI* 8(4), 890–904 (2005)
7. 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)
8. Lachaud, J.O., Vialard, A., de Vieilleville, F.: Fast, accurate and convergent tangent estimation on digital contours. *IVC* 25(10), 1572–1587 (2007)
9. Nguyen, T.P., Debled-Rennesson, I.: Parameter-free method for polygonal representation of the noisy curves. In: *Proc. of IWCIA, RPS* (2009)
10. O’Rourke, J.: An on-line Algorithm for Fitting Straight Lines between Data Ranges. *Communications of the ACM* 24(9), 574–578 (1981)
11. Reeb, G.: Sur les Points Singuliers d’une Forme de Pfaff Complètement Intégrable ou d’une Fonction Numérique. *Comptes Rendus de L’Académie ses Sciences* 222, 847–849 (1946)
12. Rodríguez, M., Largeteau-Skapin, G., Andres, E.: Adaptive Pixel Resizing for Multiscale Recognition and Reconstruction. In: *Wiederhold, P., Barneva, R.P. (eds.) IWCIA 2009. LNCS*, vol. 5852, pp. 252–265. Springer, Heidelberg (2009)
13. Debled-Rennesson, I., Feschet, F., Rouyer-Degli, J.: Optimal Blurred Segments Decomposition of Noisy Shapes in Linear Time. *Comp.& Graphics* 30, 30–36 (2006)
14. Sivignon, I., Breton, R., Dupont, F., Andres, E.: Discrete Analytical Curve Reconstruction without Patches. *IVC* 23(2), 191–202 (2005)
15. 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)
16. Vacavant, A., Coeurjolly, D., Tougne, L.: A Framework for Dynamic Implicit Curve Approximation by an Irregular Discrete Approach. *Graphical Models* 71(3), 113–124 (2009)
17. Vacavant, A.: Fast Distance Transformation on Two-Dimensional Irregular Grids. *Pattern Recognition* 43(10), 3348–3358 (2010)
18. Wenyin, L., Dori, D.: From Raster to Vectors: Extracting Visual Information from Line Drawings. *Pattern Analysis and Application* 2(1), 10–21 (1999)

Boar Spermatozoa Classification Using Longitudinal and Transversal Profiles (LTP) Descriptor in Digital Images

Enrique Alegre¹, Oscar García-Olalla¹,
Víctor González-Castro¹, and Swapna Joshi²

¹ Dep. of Electrical, Systems and Automatic Engineerings, Univ. of León, Spain

² Dep. of Electrical and Computer Engineering, Univ. of California Santa Barbara
enrique.alegre@unileon.es, sjoshi@ece.ucsb.edu

Abstract. A new textural descriptor, named Longitudinal and Transversal Profiles (LTP), has been proposed. This descriptor was used to classify 376 images of dead spermatozoa heads and 472 images of alive ones. The result obtained with this descriptor has been compared with the Pattern spectrum, Flusser, Hu, and a descriptor based on statistical values of the histogram. The features vectors computed have been classified using a back-propagation Neural Network and the kNN (k Nearest Neighbours) algorithm. Classification error obtained with LTP was 30.58% outperforming the other descriptors. The area under the ROC curve (AUC) has also been calculated confirming that the performance of the proposed descriptor is better that of the other texture descriptors.

Keywords: texture descriptors, boar semen assessment, classification, digital image analysis.

1 Introduction

In this work we have proposed a new texture descriptor in order to assess the vitality of boar semen classifying each spermatozoon head present in the samples as dead or alive. Currently, this task is accomplished using stains with fluorescence microscopy and it is impossible to fulfil the vitality assessment without this expensive equipment [13,7]. A method using phase contrast microscopes and without using stains would be very useful saving cost and time, and for that reason, we present a proposal that works with grey levels images.

The sperm assessment is a very important problem for the porcine industry. In most of the countries there is a big demand of alimentary products obtained from pig's meat, thus there are many companies trying to obtain the best pork meat at a lower price. The way to do it is by selecting the semen used in artificial insemination (AI). The AI Centres pick up only the best boar specimens and they use them in the fertilization process.

For several years the Computer-Assisted Semen Analysis (CASA) systems have been used for assessing the seminal quality [5]. Currently these systems

analyse the motility, concentration and provide some simple geometric measures of the spermatozoa's head to characterize abnormal head shapes, obtaining an assessment of the studied sample based on these values. However, there are three valuable criteria, used by veterinary experts, that these systems do not measure automatically. Those are the number and presence of proximal and distal droplets, the vitality of the sample based on the presence of dead or alive spermatozoa and the integrity of the acrosome membrane.

A number of works have addressed some of these seminal analysis problems using digital image processing. Most of them use CASA system for evaluating the relationship among the motility patterns of the sperm cells [10,4], morphology and boar fertility or for studying the sperm motility [3,5,16]. Others researches have developed new methods to characterize the sperm shape by using spectral approaches [2], or they have been looking for subpopulations [17] using shape descriptors of the spermatozoa head. There is very little work addressing the evaluation of the membrane integrity using texture descriptors [8,11], and, as far as we know, there is no work published that assess the vitality of a sample for classifying the spermatozoa heads as dead or alive.

The rest of the paper is organized as follows: section 2 explains the images acquisition followed by a brief description of the image preprocessing and segmentation. In section 3 the features vectors of classical texture descriptors used are detailed and the new proposed descriptor is explained. Section 4 indicates the classifiers being used. The obtained results for the proposed and the classical descriptors are shown in section 5, following by concluding remarks in section 6.

2 Dataset

2.1 Image Acquisition

The boar sperm images have been captured in an University of Leon spin-off called CENTROTEC. The semen samples were obtained from three different boar races-Piyorker, Large White and Landrace. Using a Nikon Eclipse microscope and a Baster A312f progressive scan camera, 450 pairs of images have been acquired. Each pair contains an image in positive phase contrast and another with two different fluorescent stains, propidium iodide (PI) that dyes dead spermatozoa as red and dichlorofluorescin (DCF) for turning green the alive spermatozoa - see figure 1. More details about the sample preparation can be found in [15].

These pairs of images have been captured in order to develop the proposed method and to test the other texture descriptors evaluated. For purpose of validation, the ground truth has been obtained using the red and green colours of the fluorescent images and later that information has been used for labelling the grey level images.

2.2 Segmentation and Preprocessing

Every spermatozoa head from the phase contrast images have been automatically segmented. First, the image regions containing the heads have been detected by

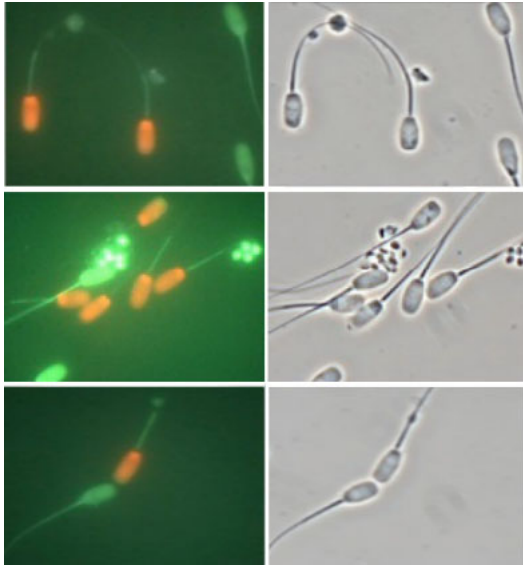


Fig. 1. Stained images using propidium iodide and dichlorofluorescein and the phase contrast one. Alive spermatozoon are coloured in red and dead ones in green. Image best viewed in color.

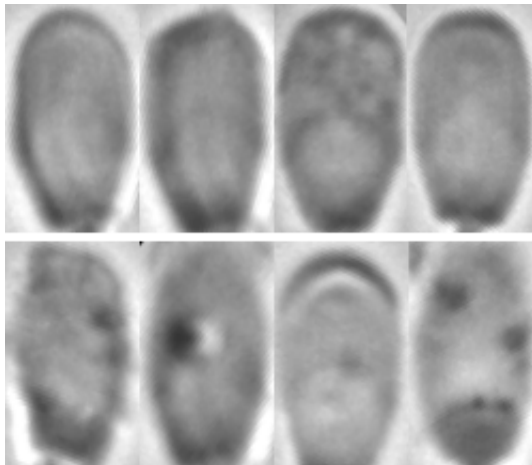


Fig. 2. Spermatozoa registered images. In the first row the alive ones and in the second row the dead ones.

thresholding and later head regions are cropped. Then, the heads have been segmented using the method explained in [9].

Later, all the spermatozoa heads have been cropped using its bounding box. The images were resized to 108×63 pixels and rotated placing the longest side in horizontal. The apical part has been placed at the right side using the location of the tail. Finally, 376 images of dead's heads and 472 images of alive's heads have been obtained. In Figure 2 we can see four registered images of alive spermatozoa in the first row and four of dead ones in the second.

3 Image Descriptors

3.1 Previous Work

Our first approach has been to try a number of classical descriptors for classifying the spermatozoa as dead or alive using the different texture distributions present in their heads. We have proposed a new descriptor, called LTP that will be explained in next section. The descriptors evaluated have been Hu, Flusser, several statistical descriptors and some variations of the Pattern Spectrum. Hu and Flusser has been tested with grey level images and with binary images. In the last case, the objective was to study if the external shape and the big dark inner regions could give enough information for distinguishing between both classes.

The following feature vectors have been used. The Hu descriptor used have been the seven normalized moments proposed by Hu [11] that are invariant to rotation, translation and scale. Furthermore, the six invariant affine moments defined by Flusser [6] have been computed. For the histogram of the cropped images we have computed four statistical features: average gray level, average contrast, measure of uniformity and entropy. All together make up the statistical feature vector used.

The Pattern Spectrum gives a histogram capturing the distribution of size of the different objects or regions present in an image. We also have computed four different Pattern Spectrum vectors using the Maragos's proposal [12], with 10 and 20 elements, both with and without normalizing the vectors.

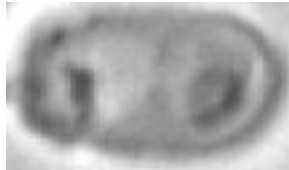


Fig. 3. A dead spermatozoa image

3.2 Our Approach: Longitudinal and Transversal Profiles (LTP) Descriptor

As can be seen in figure 2 the texture distribution in the spermatozoa alive heads is different to the dead ones. The alive ones present, in general, a smoother grey levels without big dark spots and it is possible to distinguish the acrosome membrane as an oval shape shadow in the middle of each head. We have observed the grey levels distribution along the longitudinal axis is more homogeneous in the alive ones than in dead ones. In figure 4 it is possible to see the average value of the grey level profile along the head longitudinal axis both alive spermatozoa and dead ones. Although they are similar it is possible to notice that the alive profile has two minimums, close to columns 10 and 100, with a lower grey level than the dead one. As that is not a big difference we also have considered the presence of dark spots in the spermatozoa heads as one of the main features in the dead spermatozoa heads (see fig 2). Our proposed method tries to collect that information and use it for obtaining a descriptor that provides a better classification than the others descriptors evaluated.

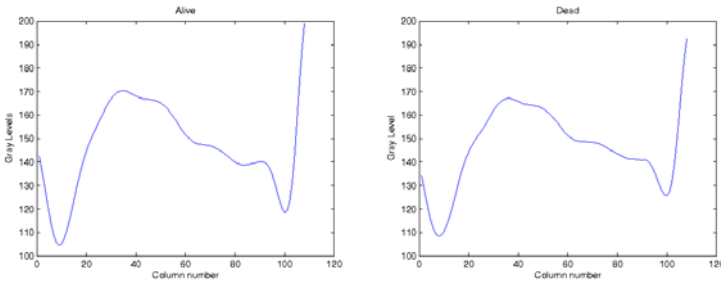


Fig. 4. In the left, the average vector of longitudinal gray scale profiles of alive spermatozoa. In the right, the same vector for dead ones.

The proposed descriptor, named LTP (Longitudinal and Transversal Profiles), gathers the grey levels along the longitudinal axis in the image middle and four transversal images profiles placed at the minimum values of the previous longitudinal profile. As we have explained in section 2.2 we are working with registered images that have been rotated and resized like the image in figure 3.

For obtaining the descriptor we do the following. First, we compute the grey level profile along the longitudinal axis, see figure 5. Then the four first minimum values of that vector are detected and the corresponding coordinates of those columns are obtained. The minimum columns are extracted as profile vectors that contain all the grey levels in each column. The four columns associated with these minimum values are concatenated as a unique row vector starting with the column with the minimum value. Later, the longitudinal profile is linked together with these row vector for making up the proposed descriptor.

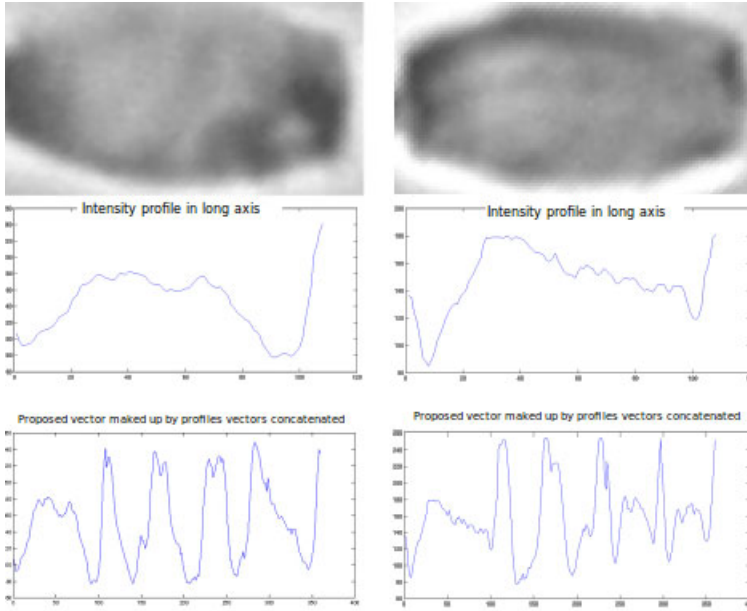


Fig. 5. An alive spermatozoa image and a dead one with their longitudinal profile and the whole features vector

The descriptor has a length of $NCols \times 4 + NRows$ where $NCols$ is the number of the columns and $NRows$ is the number of rows in the images. The first column in figure 5 shows an alive image and a dead one in the second column with, in both cases, their longitudinal profile and their features vector in second and third rows.

4 Classification

First, a kNN classification with k values 1, 3, 5, 7, 9 and 11 was carried out. The best hit rate in the most cases have been obtained with values of k greater than 7. Later, we have also classified the data using a back-propagation Neural Network with one hidden layer and a logistic sigmoid transfer function for the hidden and the output layer. Learning was carried out with a momentum and adaptive learning rate algorithm.

Data was normalized with zero mean and standard deviation equal to one. Classification was carried out by means of 10-fold cross validation with several different combinations of neurons in the hidden layer and training cycles in order to find out the optimal configuration in terms of accuracy. Some people in the machine learning community have been claiming for some years that using the hit rate to illustrate the performance of the classifier is not the most suitable option, but the ROC analysis is a more powerful tool [14]. Therefore, ROC curves

have been obtained, and so the area under them (AUC) has been calculated to perform a more powerful comparison.

5 Results

Table 1 shows the errors obtained using a kNN classifier. The table is arranged in ascending order from the global error rate. Likewise, the table 2 sums up the errors obtained using Neural network and the values of the Area Under the Curve (AUC) ROC. The table is arranged in ascending order from the global error. It is possible to observe that the errors provided by kNN do not correspond exactly with the NN results, but, in both cases, the proposed descriptor outperforms the rest. It is also interesting to notice that with the NN classifier the errors obtained with LTP are pretty balanced between classes. In general, the errors of the dead class are much bigger than the alive class. From our point of view, the reason is that the alive class images are much more homogeneous whereas the visual structure of the dead images are more scattered, harder to describe and more difficult to classify.

Table 1. Classification errors using kNN

Descriptors	Global error	Dead error	Alive error
LTP	31.94%	58.76%	11.16%
PS20	32.92%	57.06%	14.22%
PS10	33.05%	55.65%	15.54%
PS20Norm	35.64%	55.37%	20.35%
PS10Norm	36.62%	54.24%	22.98%
HuBW	37.48%	53.39%	25.16%
FlusserBW	39.33%	50.00%	31.07%
Statistical	40.00%	50.00%	33.04%
HuGray	43.03%	48.31%	38.95%
FlusserGray	44.39%	57.34%	34.35%

Table 2. Errors using neural network classifier for 200 cycles and two neurons in the hidden layer

Descriptors	AUC	Error	Error_0	Error_1	Std	Std_0	Std_1
LTP	0.7441	30.58%	39.90%	23.35%	0.8009	2.5079	1.6380
Statistical	0.7097	32.45%	47.26%	20.98%	1.3107	3.0408	1.6478
PS20	0.6836	33.43%	53.24%	18.07%	1.0678	1.5124	1.6022
PS10	0.6642	33.82%	56.51%	16.27%	0.6394	1.7614	1.4385
HuGray	0.6866	35.57%	58.88%	17.56%	1.5455	5.3074	3.0494
PS10Norm	0.6474	35.57%	57.97%	18.20%	0.4679	2.0447	1.4808
PS20Norm	0.6796	35.76%	53.58%	21.94%	1.1498	2.6397	1.4206
FlusserGray	0.6509	37.23%	67.11%	14.13%	1.7222	5.4496	1.4896
HuBW	0.5786	43.12%	87.43%	8.79%	0.5412	3.0200	2.7800
FlusserBW	0.5304	43.87%	95.19%	4.11%	0.5513	1.2854	0.6837

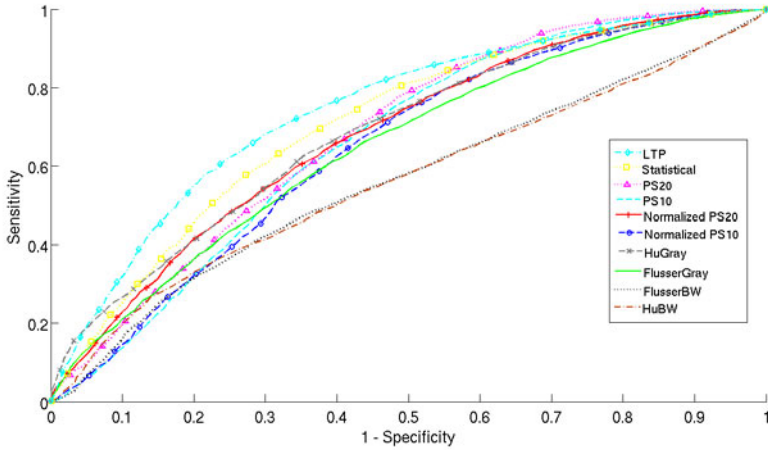


Fig. 6. ROC curve describing all the descriptors results. Image best viewed in color

The same behaviour can be seen in the ROC (Receiver Operating Characteristic) curve [6]. Although there are some descriptors with a high accuracy, as PS20, Statistical or PS10, the proposed descriptor surpasses the other ones in the interval of specificity $[0.0,0.6]$, been slightly lower to PS20 at the end of the curve.

This results, jointly with the error rates obtained with kNN and NN, allow us to say that the proposed descriptor has a better performance than the others in this context.

6 Conclusions

A new method for describing the texture present in the images of boar spermatozoa heads has been proposed. We have computed this descriptor and we have used it for classify the spermatozoa heads as dead or alive. The results obtained have been compared with several classical texture descriptor as Flusser, Hu, some variations of Pattern Spectrum and statistical descriptor composed by some statistical features, using both kNN and a back-propagation Neural Network. The results shown that the proposed descriptor outperforms the others using the error rate criteria and the accuracy seen at the ROC curve. Furthermore, the proposed descriptor has a low computational cost in front of the lot of computation required by the other evaluated techniques. An error rate of 30.58% is not enough good for presenting this method as a replacement for the manual process used currently but it do point that it is possible to assess the semen vitality with grey level images so better descriptors can be proposed in the future.

Acknowledgement

This work has been supported by grants DPI2009-08424 and PR2009-0280 from the Spanish Government. We appreciate the help of CENTROTEC for providing us the boar semen samples and for their help with the image capture. Also thanks to professor Manjunath and to all the people from the Vision Research Lab (University of California, Santa Barbara).

References

1. Alegre, E., Biehl, M., Petkov, N., Sánchez, L.: Automatic classification of the acrosome status of boar spermatozoa using digital image processing and LVQ. *Computers in Biology and Medicine* 38, 461–468 (2008)
2. Beletti, M., Costa, L., Viana, M.: A spectral framework for sperm shape characterization. *Computers in Biology and Medicine* 35(6), 463–473 (2005)
3. Betancourt, M., Resendiz, A., Casas, E., Fierro, R.: Effect of two insecticides and two herbicides on the porcine sperm motility patterns using computer-assisted semen analysis (casa) in vitro. *Reproductive Toxicology* 22, 508–512 (2006)
4. Contri, A., Faustini, C.V.M., Wegher, L., Carluccio, A.: Effect of semen preparation on casa motility results in cryopreserved bull spermatozoa. *Theriogenology* 74, 424–435 (2010)
5. Didion, B.: Computer-assisted semen analysis and its utility for profiling boar semen samples. *Theriogenology* 70(8), 1374–1376 (2008)
6. Flusser, J.: Moment invariants in image analysis. *Proceedings of the World Academy of science* 11, 196–201 (2006)
7. Gillan, L., Evans, G., Maxwell, W.: Flow cytometric evaluation of sperm parameters in relation to fertility potential. *Theriogenology* 63(2), 445–457 (2005)
8. Gonzalez, M., Alegre, E., Alaiz, R., Sánchez, L.: Acrosome integrity classification of boar spermatozoon images using dwt and texture descriptors. In: *Computational Vision and Medical Image Processing: VipIMAGE*, pp. 165–168 (2007)
9. González-Castro, V., Alegre, E., Morala-Argüello, P., Suarez, S.A.: A combined and intelligent new segmentation method for boar semen based on thresholding and watershed transform. *International Journal of Imaging* 2, 70–80 (2009)
10. Herreros, M.G., Aparicio, I.M., Núñez, I., García-Marín, L.J., Gil, M.C., Peña-Vega, F.J.: Boar sperm velocity and motility patterns under capacitating and non-capacitating incubation conditions. *Theriogenology* 63(3), 795–805 (2005)
11. Hu, M.K.: visual pattern recognition by moment invariants. *IRE Trans. Inform. Theory* 8, 179–187 (1962)
12. Maragos, P.: Pattern spectrum and multiscale shape representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 11, 701–716 (1989)
13. Mironova, E.V., Evstratova, A.A., Antonov, S.M.: A fluorescence vital assay for the recognition and quantification of excitotoxic cell death by necrosis and apoptosis using confocal microscopy on neurons in culture. *Journal of Neuroscience Methods* 163, 1–8 (2007)
14. Provost, F.J., Fawcett, T., Kohavi, R.: The case against accuracy estimation for comparing induction algorithms. In: *Proceedings of the 15th International Conference on Machine Learning*, pp. 445–453 (1998)
15. Sánchez, L., Petkov, N., Alegre, E.: Statistical approach to boar semen evaluation using intracellular intensity distribution of head images. *Cellular and Molecular Biology* 52, 38–43 (2006)

16. Saravia, F., Wallgren, M., Nagy, S., Johannisson, A., Rodríguez-Martínez, H.: Deep freezing of concentrated boar semen for intra-uterine insemination: effects on sperm viability. *Theriogenology* 63(5), 1320–1333 (2005)
17. Thurston, L., Mileham, A., Holt, W.: Morphologically distinct sperm subpopulations defined by fourier shape descriptors in fresh ejaculates correlate with variation in boar semen quality following cryopreservation. *Journal of Andrology* 22(3), 382–394 (2001)

Topology-Preserving Registration: A Solution via Graph Cuts

Lucilio Cordero-Grande, Gonzalo Vegas-Sánchez-Ferrero,
Pablo Casaseca-de-la-Higuera, and Carlos Alberola-López*

Laboratorio de Procesado de Imagen,
Escuela Técnica Superior de Ingenieros de Telecomunicación
University of Valladolid, Campus Miguel Delibes
Paseo Belén 15, 40011, Valladolid, Spain
{lcorgra,gvegsan}@lpi.tel.uva.es,
{jcasasec,caralb}@yllera.tel.uva.es
<http://www.lpi.tel.uva.es>

Abstract. In this work we propose a topology-preserving registration method based on a discrete Markov random field of deformations and a block-matching procedure. For that purpose, the fidelity of a given deformation to the data is established by a block-matching strategy, the smoothness of the transformation is favored by an appropriate prior on the field and topology preservation is guaranteed by imposing some hard-constraints on the local configurations of the field. The resulting deformation is defined as the maximum a posteriori of the field and it is estimated via graph cuts. Results on medical images show the efficiency of using graph cuts based fusion moves for the optimization of the field even though its potentials are neither sparse nor separable and the reduced fused problem turns to be non-submodular.

Keywords: image registration, topology preservation, Markov random fields, combinatorial optimization, graph cuts.

1 Introduction

Image registration is concerned with the search of an optimal transformation for the alignment between objects in two images. Let I_0 and I_1 be such a pair of images. The registration problem is mathematically defined as the computation of a spatial transformation φ so that every point in I_0 is matched to a point in I_1 . Image registration is formulated as the minimization in the space of

* This work was partially supported by the Ministerio de Ciencia e Innovación and the Fondo Europeo de Desarrollo Regional (FEDER) under Research Grants TEC2007-67073/TCM and TEC2010-17982, and by the Centro para el Desarrollo Tecnológico Industrial (CDTI) under the cvREMOD project and Research Grant CEN-20091044. The work was also funded by the Junta de Castilla y León under Grants VA027A07 and VA039A10-2, and the Consejería de Sanidad de Castilla y León under Grants GRS 292/A/08, SAN126/VA032/09 and SAN126/VA033/09.

possible transformations of an energy function which involves the intensities of the deformed and target images. That is, the registration looks for the optimal transformation φ^* that satisfies $\varphi^* = \underset{\varphi}{\operatorname{argmin}} H(\varphi)$. Usually, the energy function H takes on the form

$$H(\varphi) = \int_{\Omega} [f(I_0(\mathbf{r}), I_1(\varphi(\mathbf{r}))) + g(\varphi(\mathbf{r}))] d\mathbf{r}, \quad (1)$$

where \mathbf{r} denotes a spatial location and Ω is the integration domain of the problem. The functions $f(\cdot, \cdot)$ and $g(\cdot)$ refer respectively to a data fidelity term that accounts for the similarity between the two images for a given transformation and to a regularization term that favors the smoothness of the deformation. Generally speaking, different registration problems entail different choices for f and g and different assumptions on the transformation φ .

However, without any further requirements on the deformation field, the solution of (1) can result in unrealistic transformations. In order to improve the registration performance or to enable its use in certain applications, a proper set of constraints should be introduced in the problem. For instance, it could be interesting to obtain continuous or differentiable deformation fields. Additionally, it could be desirable to ensure the topology preservation of the transformation, which is related to its invertibility and the conservation of the connectivity and neighborhood relationships of the image objects. Finally, an unbiased registration could be guaranteed by a symmetrical energy function with respect to changes in the image roles.

Although many of the so far proposed registration methods are deterministic (i.e., no probabilistic assumptions about the energy functions are considered when posing the optimization problem), a great effort has been described in the literature on stochastic optimization procedures which, additionally, have been recently fostered by the proposal of new optimization techniques such as graph cuts (GC) or loopy belief propagation (LBP) [25]. Specifically, global solutions have been proposed very recently for the registration problem in [5].

This paper is focused on the insertion of the topology preservation constraint in a discrete stochastic formulation based on a Markov random field (MRF) of deformations and a block-matching procedure. The constraints of continuity and differentiability are also considered. As we further discuss in Section 2, our stochastic formulation of the registration problem presents some advantages with respect to other topology-preserving probabilistic approaches. Specifically, the topology preservation proposed in [1] introduces a bias in the registration solution, whereas the topology-preserving constraint established in the MRF-based approach in [8] imposes substantial restrictions to the structure of feasible configurations of the field.

As an important additional contribution of this paper, we show the viability of implementing a GC-based optimization scheme for this problem, in which the variable of the field to be optimized is vectorial, it is neither sparse nor separable and the field contains ternary potentials. We show that a fusion move [16] based on the α -expansion strategy [4] combined with the clique reduction technique

introduced in [14] and the quadratic pseudo-boolean optimization (QPBO) in [13] has been able to improve the minimum obtained by the iterated conditional modes (ICM) algorithm [3], although at the expense of a larger computational load.

In Section 2, we review the proposals that impose some of the aforementioned constraints in the registration problem. The registration methodology is described in Section 3 as well as the adopted optimization strategies. In Section 4, we present a comparative of different variants for the optimizer together with some snapshots to illustrate about the compliance of the results with the topology preservation constraint. Finally, some conclusions are presented in Section 5 together with possible extensions of the work.

2 Background

A seminal work which defines the registration problem as such of finding a homeomorphic map, i.e., a continuous bijective transformation whose inverse is also continuous, is proposed in [6]. The registration is defined as the solution of a viscous-fluid Partial Differential Equation (PDE). This formulation can deal with large deformations while simultaneously preserving the topology. The solution of the PDE is implemented by a Finite Element (FE) method, which involve a large computational load, mainly due to the successive spatial over-relaxation steps. Additionally, it is necessary to reinitialize the discretization grid to prevent for singularities in the transformation.

The Large Deformation Diffeomorphic Metric Mapping (LDDMM) framework, reviewed in [18], poses the registration problem as the computation of a geodesic path on the manifold of diffeomorphisms connecting the images. A diffeomorphism is defined as a differentiable bijective transformation with differentiable inverse. This method is formulated by a global variational problem similar to (1), but instead of regularizing the deformation field, it regularizes the temporal flow of a velocity field, which is integrated to provide the solution. One advantage of this method in comparison with [6] is that the velocity fields are smooth not only in space, but also in time. Some interesting features of this algorithm are that the solution is not only the deformation but also the path connecting the images and that it implicitly provides a measure of the distance between two images connected by a diffeomorphic map. Extensions of the basic LDDMM framework have been developed to include an affine transformation component [27], to obtain symmetric or unbiased transformations [2] or to reduce the computation requirements [11]. Diffeomorphic extensions have also been proposed for the popular Demons algorithm, with an advantage in computational efficiency [26].

On the other hand, some methods are not based on a physical process of deformation, but on its parametric representation. For instance, simple approaches based on B-splines preserve the topology by restricting the maximum deformation taking into account the spacing of the grid [23], whereas more sophisticated methods impose restrictions on the coefficients of the transformation [7].

Our proposal differs from the previous approaches in that it embodies the registration problem in a stochastic framework. Hence, our method has more in common with methodologies such as [1], where the registration solution is obtained as the maximum a posteriori (MAP) of a field of deformations whose prior favors the smoothness of the solution simultaneously ensuring one-to-one mappings and whose likelihood tends to improve the fidelity of the solution to the data. The major drawback in [1] is that the smoothness is biased with respect to the chosen diagonal direction of the triangulation [12,9]. Another stochastic approach for the registration problem is presented in [8], which formulates the registration in a discrete Bayesian framework by a MRF of deformations. However, this work establishes a simple diffeomorphic constraint by limiting the relation between the grid spacing and the maximum deformation, the same as in [23]. Specifically, diffeomorphisms are guaranteed by limiting the maximum deformation to be 0.4 times the grid spacing, which imposes an important restriction to the resolution of the field or to the range of allowed displacements. In contrast to this approach, the range of allowed displacements of our method is not restricted by the grid spacing, but by the configurations in the neighborhood of the site under consideration.

3 Registration Method

We are given a pair of zero-origin 2D discrete-space images $I_i[\mathbf{m}] = I_i(\mathbf{m}\Delta_{\mathbf{m}})$, with $i \in \{0, 1\}$ denoting each of the images and $\Delta_{\mathbf{m}}$ the pixel resolution. The registration objective is to estimate a transformation $\varphi : \Omega \rightarrow \Omega$ that maps the domain of I_0 onto the domain of I_1 . The transformation can be related with the displacement map \mathbf{u} by $\varphi(\mathbf{r}) = \mathbf{r} + \mathbf{u}(\mathbf{r})$. In our discrete formulation, the estimation is performed in a discrete set of spatial positions or *sites* \mathbf{s} , $\varphi[\mathbf{s}] = \varphi(\mathbf{s}\Delta_{\mathbf{s}})$, with $\Delta_{\mathbf{s}}$ the transformation grid resolution. The reconstruction of the continuous transformation is obtained by bilinear interpolation of the values of the transformation in the sites. Moreover, the feasible displacements take on values in the space of quantized transformations $\mathbf{u}^{\mathbf{k}} = \mathbf{k}\Delta_{\mathbf{k}}$, with $\Delta_{\mathbf{k}}$ the quantization step. The key idea of these discretizations is represented in Fig. 1.

Therefore, we search for a continuous transformation $\varphi : \Omega \rightarrow \Omega$, such that it maps the boundary of Ω onto itself and that is locally injective, that is to say

$$J(\mathbf{r}) = \left| \frac{\partial \varphi(\mathbf{r})}{\partial \mathbf{r}} \right| > 0 \quad \forall \mathbf{r} \in \Omega, \quad (2)$$

so it is a topology-preserving or homeomorphic map [19]. The continuity of the transformation is ensured by the bilinear interpolation scheme of the deformations in the grid of sites whereas the identical mapping of the boundary is ensured by establishing a zero-deformation Dirichlet boundary condition on the deformation field. The positivity of the Jacobian is somewhat more subtle and it is treated later on.

A MRF [17] is built by defining a field of discrete random variables $X^{\mathbf{s}}$ over the grid sites $\mathbf{s} \in \mathcal{S}$ which take on values in the space of quantized deformations $\mathbf{k} \in$

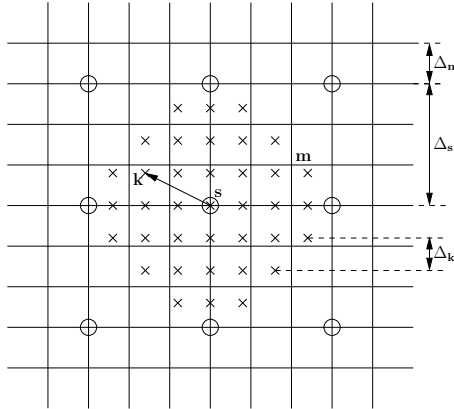


Fig. 1. Discretization of the model. Image pixels are represented as line intersections, transformation sites as circles, and transformation values for the site \mathbf{s} as crosses.

\mathcal{K} such that $u^{\mathbf{k}} \leq u_{\max}$, with u_{\max} denoting the maximum allowed deformation. The MRF is built from the definition of a system of contextual dependencies over the geometry of the sites. These dependencies are encoded in the form of a neighborhood system $\delta(\mathbf{s})$.

The Hammersley-Clifford theorem [10] states that if a Random Field is a MRF for the neighborhood δ , it is also a Gibbs Random Field (GRF) for that neighborhood and vice versa. In a GRF we have the relationship

$$\Pi(\mathbf{x}) = \frac{1}{Z} \exp(-H(\mathbf{x})), \tag{3}$$

which models the probability of occurrence of a configuration of the field $\Pi(\mathbf{x})$ as the negative exponential of the energy of that configuration $H(\mathbf{x})$ normalized by Z , the so-called *partition function*.

From a Bayesian perspective, given the image pair (I_0, I_1) , one has to consider the posterior field

$$\Pi(\mathbf{x} \mid I_0, I_1) \propto \Pi(\mathbf{x})f(I_0, I_1 \mid \mathbf{x}). \tag{4}$$

which is also Markovian and where $f(I_0, I_1 \mid \mathbf{x})$ is known as the likelihood function of the data given a configuration of the field. The definition of the posterior field is thus reduced to design a functional form for its energy function $H(\mathbf{x}; I_0, I_1)$, which is now developed.

3.1 Design of the Field

The energy function $H(\mathbf{x}; I_0, I_1)$ for the posterior field is obtained, according to (3) and (4) [17], as the sum of three potential terms which are explained below:

$$\begin{aligned}
 H(\mathbf{x}; I_0, I_1) = & \sum_{\mathbf{s} \in \mathcal{C}_1} V_1(x^{\mathbf{s}}; I_0, I_1) + \sum_{(\mathbf{s}_1, \mathbf{s}_2) \in \mathcal{C}_2} V_2(x^{\mathbf{s}_1}, x^{\mathbf{s}_2}) + \\
 & \sum_{(\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3) \in \mathcal{C}_3} V_3(x^{\mathbf{s}_1}, x^{\mathbf{s}_2}, x^{\mathbf{s}_3}).
 \end{aligned} \tag{5}$$

These terms correspond respectively to the likelihood of the data, the prior for deformation smoothness and the prior for topology preservation. The first two terms are in the spirit of (1) and the work in [8], whereas the third term is one of the contributions of our work. Note that the potential terms are defined over the *clique* categories \mathcal{C}_1 , \mathcal{C}_2 and \mathcal{C}_3 . A clique \mathcal{C} is defined as a subset of the neighborhood system δ either with a single element or in which every pair of different elements are neighbors [20], with \mathcal{C}_n denoting n -element cliques. The cliques belonging to each clique category are plotted in Fig. 2.

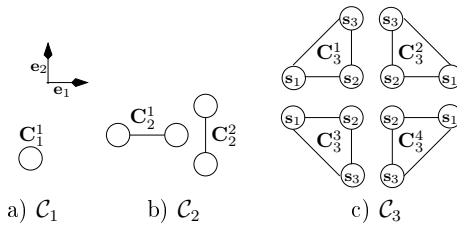


Fig. 2. Clique categories of the MRF. Note the orientation of the coordinate system $\{\mathbf{e}_1, \mathbf{e}_2\}$ such that $\mathbf{r} = r_1\mathbf{e}_1 + r_2\mathbf{e}_2$.

Henceforward we present a description of the potential terms and clarify the meaning of the cliques in Fig. 2. For that purpose, let $\mathbf{r}^{\mathbf{m}}$ denote the location of the pixel \mathbf{m} , $\mathbf{r}^{\mathbf{s}}$ the location of the site \mathbf{s} and $\mathbf{r}^{\mathbf{k}^{\mathbf{s}}}$ the transformed location that corresponds to the local configuration $x^{\mathbf{s}} = \mathbf{k}$:

$$\mathbf{r}^{\mathbf{k}^{\mathbf{s}}} = \varphi^{\mathbf{k}^{\mathbf{s}}} = \mathbf{s}\Delta_{\mathbf{s}} + \mathbf{k}\Delta_{\mathbf{k}} \qquad \mathbf{r}^{\mathbf{s}} = \mathbf{s}\Delta_{\mathbf{s}} \qquad \mathbf{r}^{\mathbf{m}} = \mathbf{m}\Delta_{\mathbf{m}}. \tag{6}$$

Finally, let continuous image values be obtained by bilinear interpolation.

Likelihood Term. For intramodality registration, the potential function V_1 could be the sum of squared differences:

$$V_1(x^{\mathbf{s}}; I_0, I_1) = \sum_{|\mathbf{m}| \leq \rho} N[\mathbf{m}](I_0(\mathbf{r}^{\mathbf{s}} + \mathbf{r}^{\mathbf{m}}) - I_1(\mathbf{r}^{\mathbf{k}^{\mathbf{s}}} + \mathbf{r}^{\mathbf{m}}))^2, \tag{7}$$

where ρ is the maximum radius of a ball that defines the block from which the image information is extracted and N is a Gaussian kernel with standard deviation $\sigma_N = \frac{\rho - 1}{2}$, so the influence of pixels with $|\mathbf{m}| > \rho$ is negligible and the contribution of those pixels closer to positions $\mathbf{r}^{\mathbf{s}}$ and $\mathbf{r}^{\mathbf{k}^{\mathbf{s}}}$ is improved. The fidelity is encoded into 1-element cliques (see Fig. 2.a), since it only depends on the local transformation as given by $x^{\mathbf{s}}$.

Smoothness Term. The smoothness term is encoded into 2-element cliques, as shown in Fig. 2b, in order to approximate the deformation gradient by finite differences of first order in both directions of the grid. The prior penalizes those deformations in which $\left| \frac{\partial \varphi}{\partial r_1} \right|$ or $\left| \frac{\partial \varphi}{\partial r_2} \right|$ are large, with r_1 and r_2 denoting the spatial coordinates $(r_1, r_2) = \mathbf{r}$:

$$V_2(x^{s_1}, x^{s_2}) = \lambda \frac{|\mathbf{r}^{\mathbf{k}^{s_2}} - \mathbf{r}^{\mathbf{k}^{s_1}}|}{|\mathbf{r}^{s_2} - \mathbf{r}^{s_1}|}, \quad (8)$$

with λ a parameter to balance the contribution of the likelihood and the prior in the field energy.

Topology-Preserving Term. As pointed out in Section 2, in contrast to [8], the topology preservation is guaranteed by an appropriate potential term in the local configurations of the field. Specifically, this term is built upon 3-element cliques, as shown in Fig. 2c. Note that the four 3-element cliques defined in Fig. 2c are analogous to the four corners from which the Jacobian is computed in [12].

In this case, the relative spatial locations of the clique elements influence the definition of the potential. Let \mathbf{s}_2 be the element in the vertex that subtends the right angle of the triangle formed by the locations of the clique elements, \mathbf{s}_1 the vertex where the non-diagonal side is along the direction \mathbf{e}_1 and \mathbf{s}_3 the vertex where the non-diagonal side is along the direction \mathbf{e}_2 (see Fig. 2c). Then, the potential term is constructed by restricting the Jacobian of the deformation in the triangle as follows

$$V(x^{s_1}, x^{s_2}, x^{s_3}) = \beta U \left(\frac{(r_1^{\mathbf{k}^{s_2}} - r_1^{\mathbf{k}^{s_1}})(r_2^{\mathbf{k}^{s_2}} - r_2^{\mathbf{k}^{s_3}}) - (r_2^{\mathbf{k}^{s_2}} - r_2^{\mathbf{k}^{s_1}})(r_1^{\mathbf{k}^{s_2}} - r_1^{\mathbf{k}^{s_3}})}{(r_1^{s_2} - r_1^{s_1})(r_2^{s_2} - r_2^{s_3})} \right), \quad (9)$$

where U is a step function such that $U(x) = 0$ if $x > 0$ and $U(x) = 1$ otherwise. Therefore, if β is large enough, the topology will be preserved. This preservation is ensured by the bilinear interpolation scheme adopted. Nevertheless, this scheme only guarantees the continuity of the transformation and its inverse. For ensuring the differentiability of the transformation, a second-order interpolation scheme is required, in which case, a larger neighborhood system would be required to simultaneously preserve the topology, with larger computational load [12].

3.2 Optimization Procedure

Once the field is completely designed, the remaining task is to establish a procedure for searching the optimal solution. This is not straightforward due to the high complexity of the field at hand. Specifically,

- The variable to be optimized is not binary, but it is vector-valued and non-separable.

- The field includes high order interactions as given by the 3-element cliques which are not sparse.

In [25], a comparison is established among the main methodologies for the optimization of MRF with smoothness-based priors. The results obtained suggest a large improvement in performance when using modern optimization techniques such as GC or LBP with respect to classical techniques. Nevertheless, the energy functions analyzed are based only on pairwise interactions between the variables of the field, so ternary interactions are not considered. Unfortunately, when dealing with ternary interactions, there are many cases in which the recent optimization techniques are not directly applicable.

Regarding LBP techniques, ternary interactions involve the introduction of a variable encoding the $|\mathcal{K}|^3$ possible configurations of the three variables, which is feasible only in those cases in which $|\mathcal{K}|$ has a small value, the ternary potentials are sparse [15] or they are separable [24]. However, our design contains vector valued non-separable ternary potentials and the condition of positivity of the Jacobian is not sparse enough, as only approximately half of the possible labels turn out to be null.

In the case of GC techniques, the problem is also intricate. The main limitation is that the reduction of the multilabel problem to a binary one by using a fusion move strategy [16] such as α -expansions [4] combined with the clique reduction technique introduced in [14] leads to a non-submodular transformed energy function. Submodularity is essential in order to obtain the respective GC optimizers in linear time and, in order to comply with it, the projections on two variables of any of the respective moves must be submodular [14], which is not our case. Nevertheless, one can still resort to the recently introduced QPBO technique [13] and trust in its ability to solve the non-submodular problem. The general procedure is illustrated in Fig. 3 and described in the subsequent paragraphs.

The α -expansion method starts with an arbitrary labeling \mathbf{x} and for each possible label $\alpha \in \mathcal{K}$ it iteratively combines two configurations \mathbf{x}_0 and \mathbf{x}_1 such that $x_0^s = x^s$ is the actual configuration and $x_1^s = \alpha$. The combination is performed by finding the minimum cut on a graph built by encoding the original energy over a binary variable which indicates if the label for each node is taken either from \mathbf{x}_0 or from \mathbf{x}_1 . Additionally, once this binarization is performed, one can reduce the 3-element potentials to a set of 2-element and 1-element potentials by introducing an auxiliary variable for each 3-element clique without affecting the energetic behavior of the field. Details on this reduction operation can be consulted in [14].

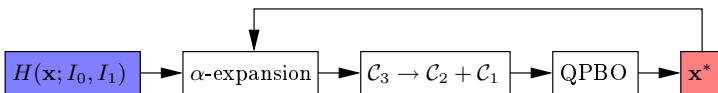


Fig. 3. Flux diagram of the GC optimization procedure

The QPBO technique is based on the reparameterization of the energy to be minimized in order to construct an equivalent graph in which the cut provides three possible states for each site $\{0, 1, \emptyset\}$, respectively for the 0-labeled, the 1-labeled and the unlabeled sites. The most important property of the label \mathbf{x} obtained by the QPBO algorithm is the *weak autarky*, that states that if \mathbf{y} is a complete labeling of the field and \mathbf{z} the label obtained by $z^s = x^s$ if $x^s = \{0, 1\}$ and $z^s = y^s$ otherwise, then $H(\mathbf{z}) \leq H(\mathbf{y})$. Taking \mathbf{y} as the global minimum of the field, this property results in the fact that the labeled pixels in \mathbf{x} are part of the global solution.

It is obvious that the QPBO algorithm will perform better if the number of unlabeled sites is small or null, which happens to be the case if the energy function is approximately or completely submodular (i.e., if it contains a small or null number of non-submodular terms). Unlabeled sites can be set to a prespecified value (in the α -expansion algorithm it is reasonable to set them to 0 in order to maintain the previous label of the site, so the energy is guaranteed not to increase with the move) or tentative labelings can be generated by suitable procedures. Specifically, in [22] two methods are proposed which extend the QPBO algorithm to cope with the unlabeled sites. The first one is the *probe* method, which pursues the exact optimum by contracting and fixing nodes on the graph and running the QPBO algorithm on the simplified graph. The second one is the *improve* method, which seeks to improve the energy of an input solution by fixing some nodes to the input solution and also running the QPBO algorithm.

4 Results

The registration algorithm has been applied to the detection of cardiac motion in a mid-cavity short axis image of the heart. The image was obtained by a cine echo-like steady-state free precession Magnetic Resonance acquisition and the end-diastolic and end-systolic phases have been extracted in order to detect the maximum cardiac motion. The parameters of the algorithm for this experiment are set to $\Delta_s = 5\Delta_m$, $\Delta_k = 2\Delta_m$, $u_{\max} = 8\Delta_m$ and $\rho = 5\Delta_m$.

Eight optimizers based on the strategies introduced in Section 3.2 are compared against the reference given by the ICM. They are configured with regard to the following features:

1. Taking the null deformation as the starting configuration (*NULL*) versus taking the ICM output as the starting configuration (*ICM*).
2. Using (*P*) or not using (\bar{P}) the probe method.
3. Using (*I*) or not using (\bar{I}) the improve method.

The results of this experiment for different values of the regularization parameter λ are included in Table II. They show that the GC-based method improves the minimum of energy obtained by *ICM*. Moreover, the insertion of the improve or probe methods tends to further improve this minimum, although they introduce a significant computational overload for small rates of energy decrease. Finally, it is not clear which starting configuration performs better, but for small values

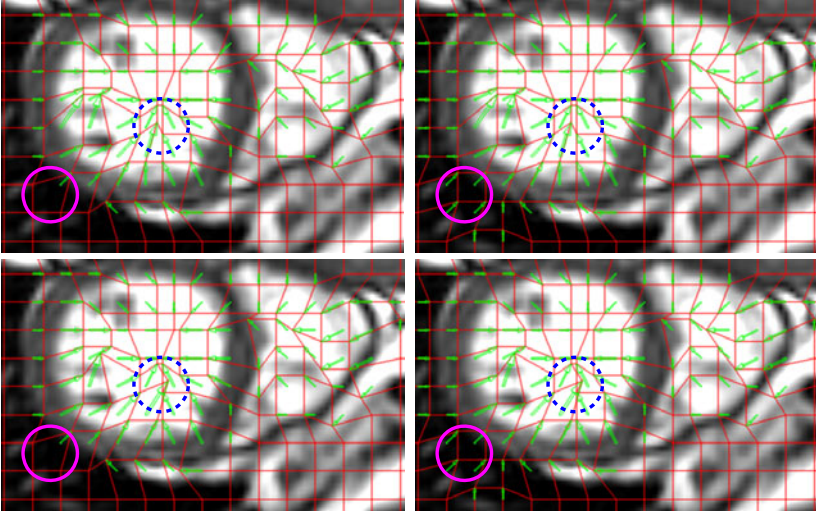


Fig. 4. Snapshots of the resulting deformation of the algorithm for $\lambda = 0.5$. Left column: *ICM*. Right column: *ICM/P/I*. Upper row: with the topology-preserving constraint. Lower row: without the topology-preserving constraint. A blue dashed circle encloses a zone in which the topology is only maintained when the constraint is introduced whereas a magenta solid circle encloses a zone in which GC-based optimization better escapes from the local minimum given by the null displacement field.

Table 1. Energy of the registration result for different optimization strategies. Computation times (in s.) are shown in parenthesis and best results for each λ are boldfaced.

Optimizer	$\lambda = 0$	$\lambda = 0.5$	$\lambda = 1$	$\lambda = 2$	$\lambda = 5$
<i>ICM</i>	327.2 (0.1)	388.2 (0.1)	435.2 (0.1)	522.5 (0.1)	812.6 (0.1)
<i>NULL/P/I</i>	352.2 (7.7)	405.1 (8.8)	430.2 (8.1)	509.3 (6.5)	732.0 (6.8)
<i>NULL/P/I</i>	328.5 (15.2)	388.1 (15.0)	429.2 (19.1)	500.9 (19.1)	729.7 (21.0)
<i>NULL/P/I</i>	331.3 (199.3)	394.0 (95.9)	430.4 (134.0)	498.9 (165.1)	729.7 (81.9)
<i>NULL/P/I</i>	328.7 (255.7)	384.2 (173.5)	429.1 (127.6)	501.0 (200.2)	729.7 (80.9)
<i>ICM/P/I</i>	326.1 (6.0)	385.6 (4.7)	432.6 (4.4)	500.3 (8.9)	729.4 (7.0)
<i>ICM/P/I</i>	325.8 (20.3)	385.5 (15.2)	432.1 (7.9)	499.3 (15.2)	729.4 (11.5)
<i>ICM/P/I</i>	325.9 (150.1)	385.5 (106.0)	428.6 (67.5)	499.3 (97.2)	729.4 (85.4)
<i>ICM/P/I</i>	325.4 (201.6)	385.5 (104.4)	428.2 (100.7)	499.3 (99.8)	729.4 (86.2)

of λ , using the *ICM* output as the starting configuration seems to improve the stability. For all these reasons, the *ICM/P/I* seems to be the best compromise between quality of the solution and computational burden.

The capability of our method to maintain the topology of the resulting deformation is illustrated in Fig. 4, which compares the behavior of the *ICM* algorithm with the GC-based *ICM/P/I* algorithm for $\lambda = 0.5$ both with and

without the topology-preserving constraint. When the constraint is removed some crosses and concavities appear in the transformation grid, which disappear when the constraint is introduced. Moreover, the GC-based optimizer seems to better escape from the local minimum given by the null displacement as noted by the large number of non-null displacements observed in this case.

5 Conclusion

This paper proposes a method to perform physically realistic registration of medical images in a stochastic discrete setting. For that purpose, we have encoded a topology preservation condition for bilinear interpolation of deformation fields into 3-element cliques. Additionally, GC techniques have been applied to effectively optimize the energy of the designed field.

Current research efforts are intended to apply the registration method to the interpolation of largely spaced tomographic images [21], to include a procedure to estimate the regularization parameter λ and to compare this new methodology for topology-preserving registration with established methods. Additionally, we plan to extend the registration method to the 3D case, which is a nontrivial task [12].

References

1. Ashburner, J., Andersson, J.L.R., Friston, K.J.: High-dimensional image registration using symmetric priors. *NeuroIm.* 9, 619–628 (1999)
2. Avants, B.B., Epstein, C.L., Grossman, M., Gee, J.C.: Symmetric diffeomorphic image registration with cross-correlation: Evaluating automated labeling of elderly and neurodegenerative brain. *Med. Image Anal.* 12, 26–41 (2008)
3. Besag, J.: On the statistical analysis of dirty pictures. *J. Roy. Stat. Soc. B. Met.* 48(3), 259–302 (1986)
4. Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell.* 23(11), 1222–1239 (2001)
5. Brown, E.S., Chan, T.F., Bresson, X.: A convex relaxation method for a class of Vector-valued Minimization Problems with applications to Mumford-Shah segmentation. Technical report. Univ. California, Los Angeles, pp. 10–43 (2010)
6. Christensen, G.E., Rabbitt, R.D., Miller, M.I.: Deformable templates using large deformation kinematics. *IEEE Trans. Image Process.* 5(10), 1435–1447 (1996)
7. Chun, S.Y., Fessler, J.A.: A simple regularizer for B-spline nonrigid image registration that encourages local invertibility. *IEEE J. Sel. Top. Signal Process.* 3(1), 159–169 (2009)
8. Glocker, B., Komodakis, N., Tziritas, G., Navab, N., Paragios, N.: Dense image registration through MRFs and efficient linear programming. *Med. Image Anal.* 12, 731–741 (2008)
9. Haber, E., Modersitzki, J.: Image registration with guaranteed displacement regularity. *Int. J. Comput. Vis.* 71(3), 361–372 (2007)
10. Hammersley, J., Clifford, P.: Markov Fields on Finite Graphs and Lattices (1971), <http://www.statslab.cam.ac.uk/~grg/books/hammfest/hamm-cliff.pdf>

11. Hernandez, M., Bossa, M.N., Olmos, S.: Registration of anatomical images using paths of diffeomorphisms parameterized with stationary vector field flows. *Int. J. Comput. Vis.* 85, 291–306 (2009)
12. Karaçalı, B., Davatzikos, C.: Estimating topology preserving and smooth displacement fields. *IEEE Trans. Med. Imag.* 23(7), 868–880 (2004)
13. Kolmogorov, V., Rother, C.: Minimizing Nonsubmodular Functions with Graph Cuts—A Review. *IEEE Trans. Pattern Anal. Mach. Intell.* 29(7), 1274–1279 (2007)
14. Kolmogorov, V., Zabih, R.: What energy functions can be minimized via graph cuts? *IEEE Trans. Pattern Anal. Mach. Intell.* 26(2), 147–159 (2004)
15. Komodakis, N., Paragios, N.: Beyond pairwise energies: Efficient optimization for higher-order MRFs. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2985–2992. IEEE Press, New York (2009)
16. Lempitsky, V., Rother, C., Roth, S., Blake, A.: Fusion moves for Markov random field optimization. *IEEE Trans. Pattern Anal. Mach. Intell.* 32(8), 1392–1405 (2010)
17. Li, S.Z.: *Markov random field modeling in image analysis*. Springer, London (2009)
18. Miller, M.I., Trounev, A., Younes, L.: On the metrics and Euler-Lagrange equations of computational anatomy. *Annu. Rev. Biomed. Eng.* 4, 375–405 (2002)
19. Musse, O., Heitz, F., Armspach, J.-P.: Topology preserving deformable image matching using constrained hierarchical parametric models. *IEEE Trans. Image Process.* 10(7), 1081–1093 (2001)
20. Paget, R., Longstaff, D.: Extracting the cliques from a neighbourhood system. *IEE Proc. Vis. Image Signal Process.* 144(3), 168–170 (1997)
21. Penney, G.P., Schnabel, J.A., Rueckert, D., Viergever, M.A., Niessen, W.J.: Registration-based interpolation. *IEEE Trans. Med. Imag.* 23(7), 922–926 (2004)
22. Rother, C., Kolmogorov, V., Lempitsky, V., Szummer, M.: Optimizing binary MRFs via extended roof duality. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8. IEEE Press, New York (2007)
23. Rueckert, D., Aljabar, P., Heckemann, R.A., Hajnal, J.V., Hammers, A.: Diffeomorphic registration using B-splines. In: *Larsen, R., Nielsen, M., Sparring, J. (eds.) MICCAI 2006. LNCS, vol. 4191*, pp. 702–709. Springer, Heidelberg (2006)
24. Shekhovtsov, A., Kovtun, I., Hlaváč, V.: Efficient MRF Deformation Model for Non-Rigid Image Matching. In: *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–6. IEEE Press, New York (2007)
25. Szeliski, R., Zabih, R., Scharstein, D., Veksler, O., Kolmogorov, V., Agarwala, A., Tappen, M., Rother, C.: A Comparative Study of Energy Minimization Methods for Markov Random Fields with Smoothness-Based Priors. *IEEE Trans. Pattern Anal. Mach. Intell.* 30(6), 1068–1080 (2008)
26. Vercauteren, T., Pennec, X., Perchant, A., Ayache, N.: Diffeomorphic demons: efficient non-parametric image registration. *NeuroIm.* 45, 61–72 (2009)
27. Younes, L.: Combining geodesic interpolating splines and affine transformations. *IEEE Trans. Med. Imag.* 15(5), 1111–1119 (2006)

Support Vector Machine Approach to Cardiac SPECT Diagnosis

Marcin Ciecholewski

Institute of Computer Science, Jagiellonian University,
ul. Łojasiewicza 6, 30-348 Kraków, Poland
marcin.ciecholewski@ii.uj.edu.pl

Abstract. This article presents the use of Support Vector Machines (SVM) to diagnose the ischemic heart disease using heart images obtained from Single Photon Emission Computed Tomography (SPECT). The data set came from 267 different patients and was divided into several sub-sets containing training and validation data. The study consisted in comparing results of classifying cardiac SPECT images using SVMs with those obtained using another method of machine learning CLIP3 which is a combination of the decision tree algorithm and the rule induction algorithm. Validations carried out using a SPECT image database have shown that SVMs are good in generalising knowledge gained about multi-dimensional data with relatively little training data.

Keywords: Support vector machine classification, cardiac SPECT imaging, medical image analysis.

1 Introduction

Every year, cardiovascular diseases kill 4.3 million people in Europe and over 2m in the European Union [1, 11]. The mortality of heart diseases is greater in Central and Eastern European countries than in Western European ones [1, 11]. Globally, the number of deaths due to cardiovascular diseases is 17.5 million, which represents 30% of all deaths every year [11]. Early detection of cardiovascular diseases plays a huge role in improving the efficiency of their treatment. If lesions are detected at an early development stage, then disease progress can be stopped, making further treatment easier. Consequently, there are more and more efforts to develop software supporting medical image analysis and helping in medical diagnostics.

Single Photon Emission Computed Tomography (SPECT) is a modern method for the non-invasive diagnostics of ischemic heart disease. The most frequent cause of this disease is atherosclerosis, caused by lipid deposits building up inside arteries, including coronary ones. The narrowed lumen of coronary arteries then causes insufficient blood flow to the heart muscle. These stenoses can be identified precisely during coronarography, while SPECT supports the assessment of the degree to which they impair blood supply to the heart muscle. If a SPECT examination shows regular blood supply, ischemic heart disease can be

ruled out with a high probability and the invasive coronarography examination can be avoided. SPECT imaging can also help in selecting the artery in which patency can be restored with a balloon or in which a bypass should be grafted. The examination is performed twice: one time at rest, and the second time after a workload stress or a pharmacological stress. After a radioactive agent is injected intravenously, a series of images is acquired using detectors moving around the patient and collecting photons emitted in different directions by that tracer. A 3D image of the heart muscle can be reconstructed from the acquired images using the right algorithms. However, cardiologists most frequently assess the images of 2D slices (cross-sections) along several different planes, like in Fig. 1.

The bright areas in these images correspond to parts of the heart muscle well supplied with blood, while if an area corresponding to a given fragment of the heart muscle is dark, it is ischemic, see Fig 2. Individual images of slices can be split into regions (Fig. 3). A cardiologist assesses every one of the 22 regions based on his/her knowledge of the appearance of a SPECT image of a healthy human heart and compares the appearance of regions in the rest state and under stress. Based on these partial diagnoses for particular regions, the final diagnosis is made. The purpose of this study was to develop an application based on Support Vector Machines (SVM) [17, 18] to classify SPECT heart images.

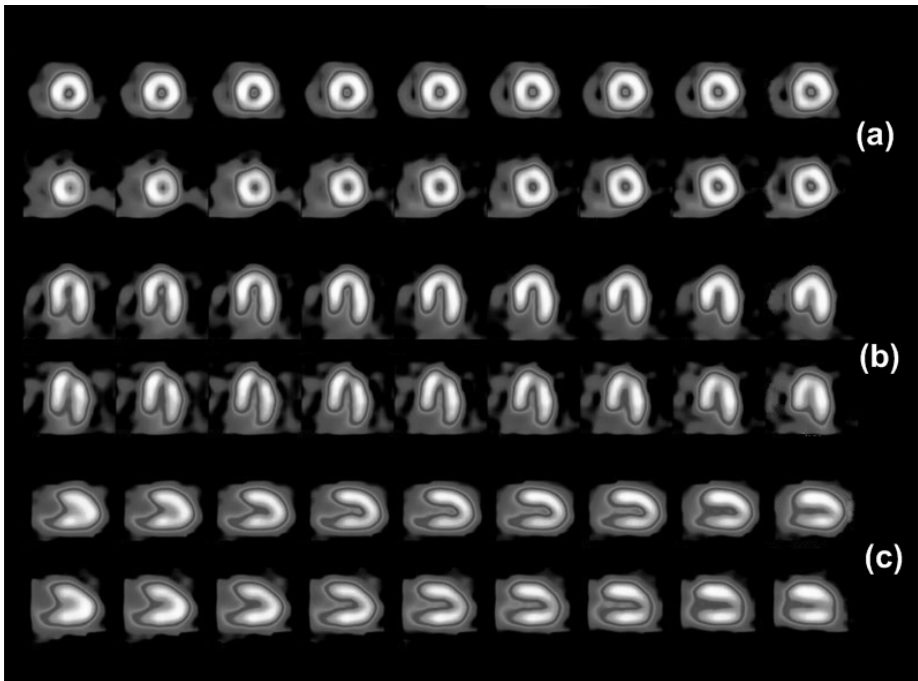


Fig. 1. Series of SPECT heart images in various cross-sections: (a) short axis of the heart (perpendicular to the long axis), (b) vertical long axis, (c) horizontal long axis of the heart

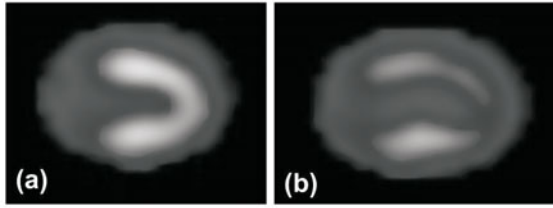


Fig. 2. Perfusion on cardiac SPECT images, (a) normal perfusion, (b) abnormal perfusion

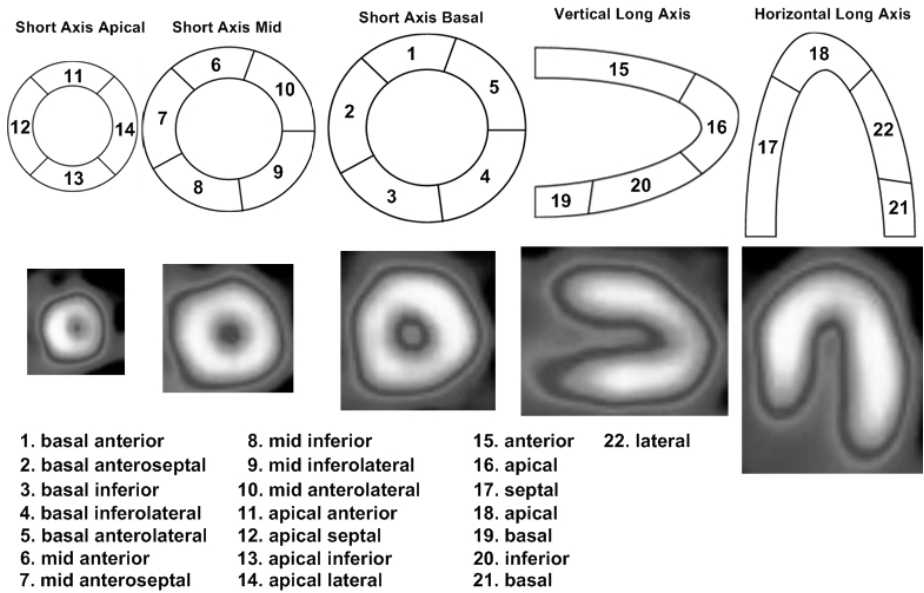


Fig. 3. Heart cross-section regions and their corresponding SPECT images of cross-sections

The particular purpose of this study was to compare the results of classifying cardiac SPECT images using SVMs with results obtained in [12] using other methods. SVM was already used for classifying SPECT images to differentiate between images from healthy subjects and images from Alzheimers disease (AD) patients. The proposed methodology yielded the accuracy greater than 90% in the diagnosis of the AD [16] and achieved the sensitivity of 84.4% at 90.9% specificity [9]. Results described in paper [16] were obtained using a linear kernel function and a contiguous linear SVM [9], respectively. In this work, the Gaussian kernel function (see tab. 2) provided the best results, i.e. 92.31% accuracy.

The traditional approach, where artificial neural networks are used for classifying, can frequently yield results not good enough, due to overtaining, which has a negative impact on the ability of the learning machine to generalise the

knowledge gained [10]. It is precisely due to the very good knowledge generalisation capability that Support Vector Machines (SVMs) is used more and more frequently [17, 18]. SVMs are good in generalising the knowledge gained, even for multi-dimensional data with relatively little training data. In addition, SVM performance is good even without any a priori knowledge of the problem under consideration: for instance, in many applications of learning machines to recognising images, the same unrestricted permutation of pixels in all images does not change the training results for SVMs, but it does for neural networks, however well they had worked previously [2].

Publication [12] deals with classifying cardiac SPECT images using a CLIP3 algorithm, which is a combination of the decision tree algorithm C4.5 [15] and rule induction algorithm CN2 [7]. The CLIP3 algorithm has already been used to classify cardiac SPECT images and its performance was comparable to that of learning algorithms C4.5 and CN2 [6].

The application under development, which uses SVMs, has been validated on the same database of cardiac SPECT images [8] as the CLIP3 method [12]. The data set came from 267 different patients. The experiments conducted have shown that the use of SVMs yielded better classification results than the CLIP3 method. The content of this paper is laid out as follows: section 1 gives the method of classification using support vectors, section 2 presents the analyzed data, section 3 discusses the experiments conducted and selected research results. And finally, the conclusions are drawn in section 4.

2 Support Vector Classification

Lets consider data in the form of vectors from space R^n belonging to two classes, to which we respectively assign numbers from set $\{-1, 1\}$:

$$D = \{(x_1, y_1), \dots, (x_n, y_n), x_i \in R^n, y_i \in \{-1, 1\}, i = 1 \dots m\} \quad (1)$$

The support vector machine algorithm separates classes of input patterns with the hyperplane for which the distance between the nearest vectors of these two classes given by relationship (1) is the greatest. This hyperplane is called the optimal separating hyperplane. It is defined as follows:

$$H : f(x) = \langle w, x \rangle + b \quad (2)$$

where x is a vector from the R^n space, w is the vector perpendicular to hyperplane H , and the value $\frac{b}{\|w\|}$ determines the offset from the beginning of the coordinate system. Fig. 4 shows an example of two different vector classes and a hyperplane separating them.

Finding the optimal hyperplane for which the distance of the nearest vectors from two different classes is the greatest is equivalent to minimising the square of the following norm [10]: $\frac{1}{2}\|w\|^2$. The case of linear data separation can be generalised to data separation with a non-linear function. For this purpose, data vectors from the R^n space are mapped to a new Hilbert space H using a certain

mapping function $\phi : R^n \rightarrow H$. As the data appears only in the form of scalar products, the form of the ϕ function does not have to be given explicitly, the following K kernel function can be used

$$K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j) \tag{3}$$

Points belonging to two classes (1) cannot be linearly separable, as shown in Fig. 4. In such cases, one of the solutions may be to slacken the condition that there must be no points between pattern classes. For this purpose, slack variables $\xi \geq 0$ are introduced. The constraining conditions will then take the following form:

$$y_i \cdot [\langle w, x_i \rangle + b \geq 1 - \xi_i], \quad \xi_i \geq 0, \quad i = 1 \dots n \tag{4}$$

where $y_i \in \{-1, 1\}$ denotes the appropriate labels of classes of input patterns x_i in accordance with the relationship (1). As a result, the SVC classification is equivalent to a minimisation problem with the following objective function:

$$\frac{1}{2} \|w\|^2 + C \left[\sum_{i=1}^n \xi_i \right]^m \tag{5}$$

where $m = 1$ is usually assumed. Constant C evaluates the amount of the penalty for the wrong classification of training data.

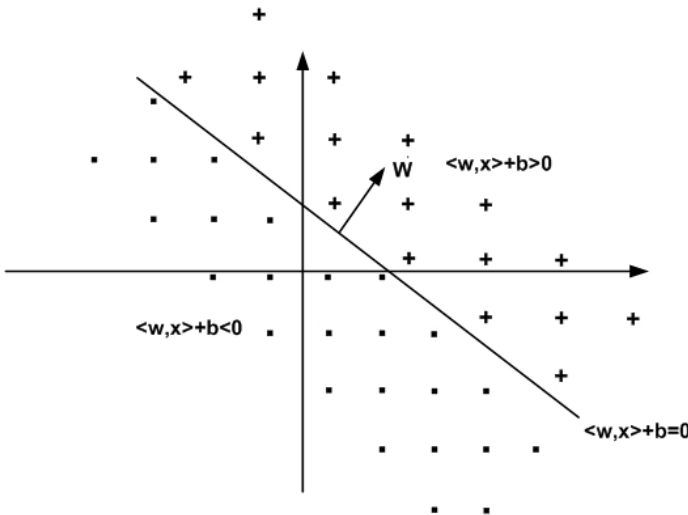


Fig. 4. The optimal hyperplane separating two vector classes. In dot-product space the hyperplane can be specified as a set of vectors x that satisfies $\langle w, x \rangle + b = 0$. Vector w is perpendicular to the plane, while the scalar value $\frac{b}{\|w\|}$ represents the shift relative to the beginning of the coordinate system.

Using the Lagrange multipliers technique the optimisation problem formulated above can be transformed as follows:

$$L_D(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j K(x_i, x_j) \quad \text{and} \quad L_D(\alpha) \rightarrow \min \quad (6)$$

under the conditions: $0 \leq \alpha_i \leq C$ and $\sum_{i=1}^n \alpha_i y_i = 0$

In the above relationship, vector $\alpha = [\alpha_1, \alpha_2, \dots, \alpha_n]$ defines non-negative Lagrange coefficients. It should be noted that separating the pattern classes makes it possible to use the kernel function $K(., .)$. This is possible, because mapping ϕ satisfies equation (3).

Non-negative Lagrange coefficients which constitute solutions to equation (6) may be used in the following decision-making function:

$$f(x) = \sum_{i=1}^n \alpha_i y_i K(x_i, x) + b \quad (7)$$

whereas:

$$b = y_i - \sum_{j=1}^n \alpha_j y_j K(x_j, x_i) \quad (8)$$

The solution to equation (4) can be determined using general quadratic programming methods. In addition, dedicated heuristic methods [13, 14] have been developed which effectively solve many classes of problems.

The optimisation problem given by relationship (6) is a Convex Quadratic Programming Problem [3]. Consequently, the use of the SVC method has an advantage over neural networks for which the occurrence of many local minima may make the learning process complex, frequently leading to poor classification.

3 Analyzed Data

Specially processed images from SPECT examinations of 267 patients were used for the analysis. Every examination consisted of images of five heart slices for the rest study and five for the stress study. This yields the total number of 44 regions for every examination - 22 for each study. For every region Ω the following value was calculated:

$$f_i = \frac{\sum_{(m,n) \in \Omega} J(m,n)}{N(\Omega)} \cdot \frac{100\%}{M} \quad (9)$$

where $J(m, n)$ denotes the brightness of the point with the coordinates (m, n) , $N(\Omega)$ denotes the number of points belonging to Ω , and M is the greatest brightness of a point from the Ω area during the rest and stress studies. Thus, the database used [8] contained 267 sets of 44 values calculated based on (9). Every set of these 44 values had a corresponding cardiologists diagnosis a healthy

or pathological perfusion of the heart muscle. In addition, a database of partial diagnoses was also used. It contained 267 sets of 22 partial diagnoses each and the appropriate final diagnosis as in the previous case:

$$\begin{aligned} \text{data sets } DF_1 \text{ and } DF_2 : & \quad DF, f_1, f_2, \dots, f_{44} \\ \text{data sets } DPD1 \text{ and } DPD2 : & \quad DF, DPart_1, DPart_2, \dots, DPart_{22} \end{aligned}$$

where DF - final diagnosis, $DPart_i$ - i^{th} partial diagnosis

Every set contained 55 cases diagnosed as healthy heart muscle perfusions (positive patterns) and 212 cases diagnosed as pathological perfusions (negative patterns). The data was divided into training and validation sets in accordance with the table below.

Table 1. Division of data into training and validation sets

	Training data set		Validation data set		Total
	Num. of pos. examples	Num. of neg. examples	Num. of pos. examples	Num. of neg. examples	
DPD1	40	40	15	172	267
DPD2	40	162	15	50	267
DF1	40	40	15	172	267
DF2	40	162	15	50	267

4 Completed Experiments and Selected Research Results

SVMs analyzed patterns from the training set, one after another for the bases shown in table 1. Then, the patterns not previously used from the validation set were employed to validate the ability of the SVMs to generalise. As finding a lesion is treated as a negative diagnosis, while the lack of lesions as a positive one, four possible values determine the classifiers behaviour. These are as follows: (TN) true negative, (FP) false positive, (FN) false negative and (TP) true positive. For each validation set and specific parameters of the SVM, the specificity, sensitivity and accuracy were calculated according to the formulas below:

$$Sensitivity = \frac{TP}{TP + FN} \cdot 100\% \quad (10)$$

$$Specificity = \frac{TN}{TN + FP} \cdot 100\% \quad (11)$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \cdot 100\% \quad (12)$$

Sensitivity is the measure of the number of correctly classified positive patterns from the validation set. Specificity is the measure of the number of correctly classified negative patterns from the validation set. Accuracy is the measure of the number of correctly classified patterns in the entire validation set.

Table 2. Selected kernel functions

Kernel function	Formula
Polynomial	$K(x, y) = (x \cdot y)^d$
Modified polynomial	$K(x, y) = (x \cdot y + 1)^d$
Sigmoid	$K(x, y) = \tanh(\rho x \cdot y + \eta)$
RBF - radial base function	$K(x, y) = \exp\left(-\frac{\ x-y\ }{2\sigma^2}\right)$
GRBF - Gaussian function	$K(x, y) = \exp\left(-\frac{\ x-y\ ^2}{2\sigma^2}\right)$

Table 3. Specificity, sensitivity and accuracy of various data sets

Data Set	Specificity		Sensitivity		Accuracy	
DF2	60.00%	9/15	94.00%	47/50	86.15%	56/65
DF1	73.33%	11/15	77.91%	134/172	77.54%	145/187
DPD2	73.33%	11/15	98.00%	49/50	92.31%	60/65
DPD1	80.00%	12/15	80.81%	139/172	80.75%	151/187

After trial validations of kernel functions presented in Table 2, including polynomial functions of various degrees, SVMs with a Gaussian kernel function were selected due to their better learning results.

For every base from Table 1, validations were conducted and parameters defined by equations 10–12 were calculated for various values of the σ variable of the Gaussian kernel function. The results are presented in Table 3 and graphs Fig. 5 - Fig. 8. Parameter C of the SVM was in every case selected so as to optimise the training results. Figures Fig. 5 - Fig. 8 show the dependence of

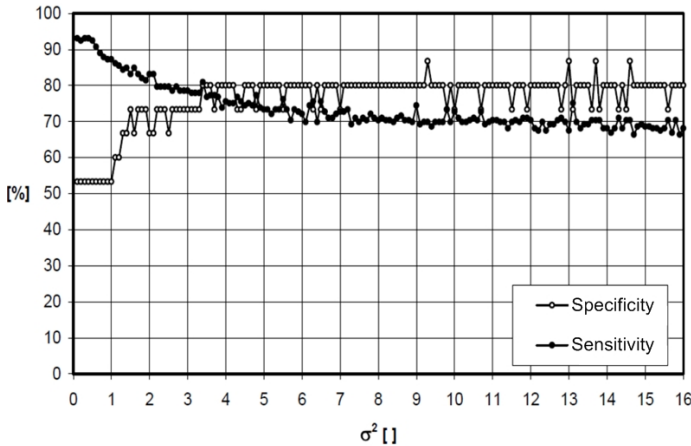


Fig. 5. The dependence of the specificity and sensitivity on the σ^2 parameter of the SVM. Results for the validation set DPD1 and parameter C = 940.

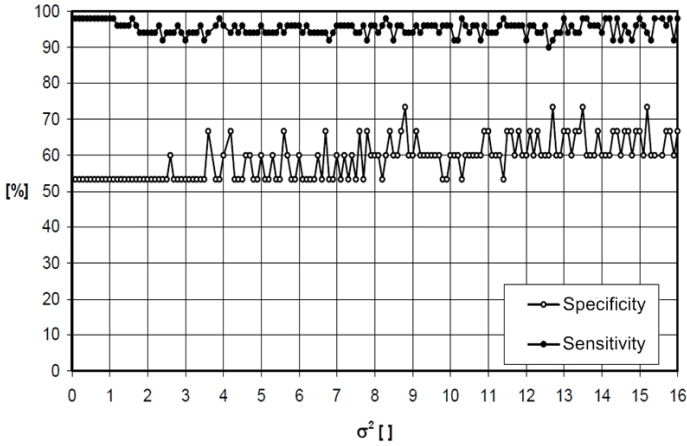


Fig. 6. The dependence of the specificity and sensitivity on the σ^2 parameter of the SVM. Results for the validation set DPD2 and parameter $C = 10000$.

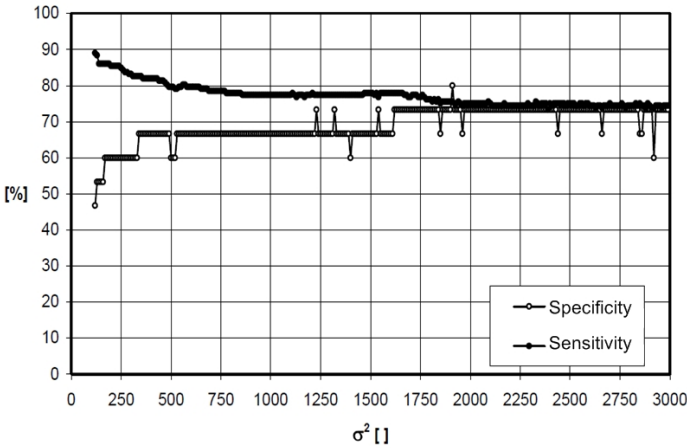


Fig. 7. The dependence of the specificity and sensitivity on the σ^2 parameter of the SVM. Results for the validation set DF1 and parameter $C = 10000$.

sensitivity and specificity on the σ parameter value. Table 3 compiles the values of sensitivity, specificity and accuracy for optimum values of parameters of the SVM. It should be noted that due to a large difference between the number of positive and negative validating patterns, accuracy is not the value which should determine the choice of SVM parameters as optimum. A better criterion would, for instance, be the mean of sensitivity and specificity. Consequently, the values in Table 3 correspond to the maximum mean value of sensitivity and specificity. Obtaining a high value of sensitivity only or specificity only is obviously not a

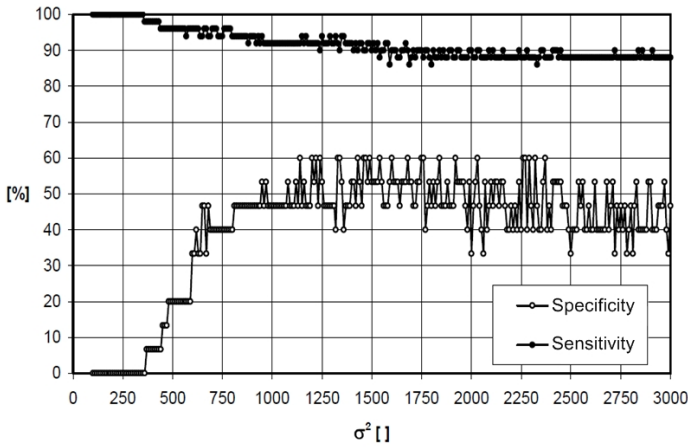


Fig. 8. The dependence of the specificity and sensitivity on the σ^2 parameter of the SVM. Results for the validation set DF2 and parameter $C = 10000$.

Table 4. A comparison of results for SVM and CLIP3

Data Set	Specificity		Sensitivity		Accuracy	
	SVM	CLIP3	SVM	CLIP3	SVM	CLIP3
DF2	60.00%	66.67%	94.00%	88.00%	86.15%	83.08%
DF1	73.33%	73.33%	77.91%	77.32%	77.54%	77.00%
DPD2	73.33%	66.67%	98.00%	74.00%	92.31%	72.31%
DPD1	80.00%	73.33%	80.81%	80.23%	80.75%	79.68%

Table 5. A comparison of best results

Specificity		Sensitivity		Accuracy	
SVM	CLIP3	SVM	CLIP3	SVM	CLIP3
73.33%	80.00%	98.00%	84.30%	92.31%	83.96%

good indicator of the effectiveness of the training machine, so the values of both these variables should be taken into account.

5 Conclusion

Both for training data in the form of partial diagnoses (data sets DPD1 and DPD2) and for data in the form of a set of SPECT image parameters (set DF2), the diagnosis using SVMs is obviously highly accurate ($>80\%$). In the case of set DF1, accuracy is only slightly below 80% . Diagnosis sensitivity and specificity

are also high, only for set DF2 does the specificity not exceed 70%. Sensitivity is particularly high (>90%) for sets DF2 and DPD2, for which there is a large set of positive training patterns (172 patterns). The difference in sensitivity for sets DF1 and DF2 as well as DPD1 and DPD2 is significant: over 16%. This suggests that the number of positive training patterns (40 patterns) in the case of DF1 and DPD1 bases is too small to fully utilise the capabilities of SVMs. A similar conclusion comes to mind for the number of negative training patterns (40 patterns) for all bases. The significant difference between the specificity and sensitivity for bases DF2 (34%) and DPD2 (~25%) is the consequence of a significant difference between the number of positive and negative training patterns. In addition, the small number of negative validation patterns means that it is difficult to assess the specificity of SVM-based diagnosis well, as a difference in the correct classification of one validating pattern produces a change in specificity exceeding 6%.

Table 4 compares results of SPECT image classification with the use of SVMs and of the CLIP3 algorithm, which is a combination of the decision tree algorithm and the rule induction algorithm. Values of validations which are higher for SVMs than for CLIP3 have been printed in bold font in table 4. It is clear that only in one case is the value lower for SVMs than for CLIP3 (specificity for DF2, difference in only one pattern). Also in only one case (specificity for DF1) were the values obtained for SVM and CLIP3 equal. In all other cases, SVMs performed better. The greatest difference is visible for the DPD2 set, where the difference between SVMs and CLIP3 is 24% for sensitivity and 20% for accuracy. In [12] the authors have presented an improved version of the CLIP3 algorithm. They obtained better results, but only for the DPD1 data set (80% specificity, 84.30% sensitivity and 83.96% accuracy). The results are presented in table 5 together with the best result for SVMs (for the DPD2 set). Here, the specificity is lower for SVM, but this is due to a difference in the classification of just one pattern. This is why, taking into account the significantly better sensitivity and accuracy results of SVMs, the best result for the support vector machines seems to be better than the best result obtained using the CLIP3 algorithm. It should be noted that the results obtained were influenced by the correct interpretation of SPECT images by physicians [12]. The diagnoses made using 41 SPECT images which were taken to create base [8] were formulated by various cardiologists, and depending, *inter alia*, on their experience, the diagnoses may have been more or less correct. Regardless of these potential errors, cardiologists diagnoses were used as the point of reference both in [12] and in this publication.

Acknowledgements

This research was financed with state budget funds for science for 2009-2012 as research project of the Ministry of Science and Higher Education: N N519 406837.

References

1. Allender, S., et al.: European Cardiovascular Disease Statistics 2008 edition, European Heart Network (2008)
2. Burges, C.J.C.: A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery* 2(2), 955–974 (1998)
3. Burges, C.J.C., Crisp, D.: Uniqueness of the SVM solution. In: *Advances in Neural Information Processing Systems*, vol. 12, pp. 223–229. MIT Press, Cambridge (1999)
4. Ciecholewski, M.: Gallbladder Segmentation in 2-D Ultrasound Images Using Deformable Contour Methods. In: Torra, V., Narukawa, Y., Daumas, M. (eds.) *MDAI 2010. LNCS (LNAI)*, vol. 6408, pp. 163–174. Springer, Heidelberg (2010)
5. Ciecholewski, M.: Gallbladder Boundary Segmentation from Ultrasound Images Using Active Contour Model. In: Fyfe, C., Tino, P., Charles, D., Garcia-Osorio, C., Yin, H. (eds.) *IDEAL 2010. LNCS*, vol. 6283, pp. 63–69. Springer, Heidelberg (2010)
6. Cios, K.J., Pedrycz, W., Swiniarski, R.: *Data Mining Methods for Knowledge Discovery*. Kluwer, Dordrecht (1998)
7. Clark, P., Niblett, T.: The CN2 algorithm. *Machine Learning* 3, 261–283 (1989)
8. Data Bases SPECT i SPECTF from UCI Machine Learning Repository, on-line <http://www.ics.uci.edu/~mlearn/MLRepository.html>
9. Fung, G., Stoeckel, J.: SVM feature selection for classification of SPECT images of Alzheimer’s disease using spatial information. *Knowledge and Information Systems* 11(2), 243–258 (2007)
10. Gunn, S.: Support vector machines for classification and regression. Technical Report, Dept. of Electronics and Computer Science, University of Southampton, Southampton, U.K (1998)
11. International Cardiovascular Disease Statistics 2009, Statistical Fact Sheet – Populations, American Heart Association (2009)
12. Kurgan, L.A., Cios, K.J., et al.: Knowledge discovery approach to automated cardiac SPECT diagnosis. *Artificial Intelligence in Medicine* 23(2), 149–169 (2001)
13. Osuna, E., Freund, R., Girosi, F.: An improved training algorithm for support vector machines. In: Principe, J., Gile, L., Morgan, N., Wilson, E. (eds.) *Neural Networks for Signal Processing VII*, pp. 276–285. IEEE, New York (1998)
14. Platt, J.: Fast training of support vector machines using sequential minimal optimization. In: Schölkopf, B., Burges, C.J., Smola, A.J. (eds.) *Advances in Kernel Methods – Support Vector Learning*, pp. 185–220. MIT Press, Cambridge (1999)
15. Quinlan, J.R.: *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Francisco (1993)
16. Salas-Gonzalez, D., Górriz, J.M., et al.: Selecting Regions of Interest in SPECT Images Using Wilcoxon Test for the Diagnosis of Alzheimers Disease. In: Graña Romay, M., Corchado, E., Garcia Sebastian, M.T. (eds.) *HAIS 2010. LNCS*, vol. 6076, pp. 446–451. Springer, Heidelberg (2010)
17. Vapnik, V.: *The Nature of Statistical Learning Theory*. Springer, New York (1995)
18. Vapnik, V.: *Statistical Learning Theory*. John Wiley & Sons, Inc., New York (1998)

An Entropy-Based Technique for Nonrigid Medical Image Alignment

Mohammed Khader and A. Ben Hamza

Concordia Institute for Information Systems Engineering
Concordia University, Montréal, QC, Canada

Abstract. In this paper, we propose a nonrigid image registration technique by minimizing an information-theoretic measure using the quasi-Newton method as an optimization scheme and a cubic B-spline for modeling the nonrigid deformation field between the reference and target 3D image pairs. Experimental results are provided to demonstrate the registration accuracy of the proposed approach. The feasibility of our method is demonstrated on a 3D magnetic resonance data volume.

Keywords: Image registration, entropy, nonrigid.

1 Introduction

In recent years, a number of intensity-based techniques have been proposed to tackle the nonrigid image registration problem [6]. A general framework for these methods relies on entropic measures. One such similarity measure is the mutual information (MI), proposed independently by Viola and Wells [17] and by Maes *et al.* [9], which has been effective in the development of the intensity-based image registration because of its ability to register images from different modalities [10,1]. Registration algorithms that maximize MI over rigid and affine transformations have reported impressive registration results. Rueckert *et al.* [14] presented MI-based schemes for matching multimodal image pairs using B-splines by representing the deformation field on a regular grid. Most accurate methods for nonrigid image registration are inspired by models from physics, either from elasticity [4,5], or fluid mechanics [2,3] but they are considered computationally expensive. Hence, several methods have been proposed based on various heuristics to approximate the underlying physical reality by alternative mathematical models [1]. Likar and Pernus [7] proposed a hierarchical image subdivision strategy by decomposing the nonrigid matching problem into an elastic interpolation of various local rigid registrations of sub-images of decreasing size. This algorithm is applicable to both intra- and inter-modal cases as it maximizes the local MI among sub-images. Although MI has been successfully applied to nonrigid image registration, it is worth noting that the MI-based registration methods might have a limited performance, once the initial misalignment of the two images is large or equally the overlay region of the two images is relatively small. Moreover, MI is sensitive to the changes that occur in

the distributions (overlap statistics) as a result of changes in the region of overlap. To circumvent these limitations, various approaches have been proposed to improve the robustness of MI-based registration, including normalized mutual information (NMI) [15]. The NMI approach is a robust similarity measure that allows for fully automated intermodal image registration algorithms. Moreover, the NMI-based registration is less sensitive to the changes in the overlap of two images. Wang and Vemuri [19] introduced the cross-cumulative residual entropy (CCRE), which is a measure of entropy defined using cumulative distributions. In this approach, CCRE between two images to be registered is maximized over the space of smooth and unknown nonrigid transformations. The reported results showed a better performance than MI-based methods. In [8], Loeckx *et al.* proposed the conditional mutual information (cMI) as a new similarity measure for nonrigid image registration. This measure was calculated as the expected value of the cMI between the image intensities given the spatial distribution. Recently, Myronenko *et al.* [13] proposed to minimize a residual complexity (RC) instead of mutual information. This approach deals with complex spatially-varying intensity distortions and produces accurate registration results.

In this paper, we propose a nonrigid image registration approach by optimizing the Jensen-Tsallis (JT) similarity measure using the quasi-Newton method as an optimization scheme and a cubic B-spline for modeling the nonrigid deformation field between the reference and target 3D image pairs. The analytical gradient of the JT similarity is derived so that we can achieve an efficient and accurate nonrigid registration.

The rest of the paper is organized as follows. In Section 2, we describe in detail the proposed method. Section 3 provides experimental results on a medical imaging dataset that demonstrate the effectiveness and superior performance of our method compared to RC and NMI approaches. And finally, in Section 4, we conclude and point out future work directions.

2 Proposed Method

2.1 Jensen-Tsallis (JT) Similarity

Shannon’s entropy of a probability distribution $\mathbf{p} = (p_1, p_2, \dots, p_k)$ is defined as $H(\mathbf{p}) = -\sum_{j=1}^k p_j \log(p_j)$. A generalization of Shannon entropy is Tsallis entropy given by

$$H_\alpha(\mathbf{p}) = \frac{1}{1-\alpha} \left(\sum_{j=1}^k p_j^\alpha - 1 \right) = -\sum_{j=1}^k p_j^\alpha \log_\alpha(p_j), \tag{1}$$

where \log_α is the α -logarithm function defined as $\log_\alpha(x) = (1-\alpha)^{-1}(x^{1-\alpha} - 1)$ for $x > 0$, and $\alpha \in (0, 1) \cup (1, \infty)$ is the entropic index. This generalized entropy is widely used in statistical physics applications [16].

Definition 1. Let $\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_n$ be n probability distributions. The Jensen-Tsallis (JT) divergence is defined as

$$D_\alpha^\omega(\mathbf{p}_1, \dots, \mathbf{p}_n) = H_\alpha \left(\sum_{i=1}^n \omega_i \mathbf{p}_i \right) - \sum_{i=1}^n \omega_i H_\alpha(\mathbf{p}_i). \tag{2}$$

where $H_\alpha(\mathbf{p})$ is Tsallis entropy, and $\omega = (\omega_1, \omega_2, \dots, \omega_n)$ is a weight vector such that $\sum_{i=1}^n \omega_i = 1$ and $\omega_i \geq 0$.

Using the Jensen inequality, it is easy to check that if $\alpha > 0$ then the JT divergence is nonnegative, symmetric and vanishes if and only if all the probability distributions are equal. Moreover, if $\alpha \in [1, 2]$ then the JT divergence D_α^ω is a convex function [12]. In the sequel, we will restrict $\alpha \in [1, 2]$, unless specified otherwise. In addition to its convexity property, the JT divergence is an adapted measure of disparity among n probability distributions as shown in the next result [12].

Proposition 1. The JT divergence achieves its maximum value when the probability distributions $\mathbf{p}_1, \dots, \mathbf{p}_n$ are distributions, that is $\mathbf{p}_i = (\delta_{ij})$, where $\delta_{ij} = 1$ if $i = j$ and 0 otherwise.

Proposition 2. The upper-bound of D_α^ω is given by $D_\alpha^\omega(\mathbf{p}_1, \dots, \mathbf{p}_n) \leq H_\alpha(\omega)$.

Since $H_\alpha(\omega)$ attains its maximum value when the weights are uniformly distributed (i.e. $\omega_i = 1/n, \forall i$), it follows that a tight upper bound of the JT divergence is given by

$$D_\alpha^\omega(\mathbf{p}_1, \dots, \mathbf{p}_n) \leq H_\alpha(1/n, \dots, 1/n) = \log_\alpha n \tag{3}$$

If we are measuring the similarity, $S_\alpha^\omega(\mathbf{p}_1, \dots, \mathbf{p}_n)$, between probability distributions, then using Eq. (3) we may define the JT similarity measure as follows:

$$S_\alpha^\omega(\mathbf{p}_1, \dots, \mathbf{p}_n) = 1 - \frac{D_\alpha^\omega(\mathbf{p}_1, \dots, \mathbf{p}_n)}{\log_\alpha n}. \tag{4}$$

Fig. 1 illustrates the JT similarity between two Bernoulli distributions $\mathbf{p} = (p, 1 - p)$ and $\mathbf{q} = (1 - p, p)$ for different values of the entropic index. As shown in Fig. 1, the highest similarity corresponds to the entropic index $\alpha = 2$.

2.2 Problem Statement

In the sequel, we will use the JT similarity measure as a matching criterion to solve the image alignment problem. Let I and J be two misaligned images to be registered, where I is the reference image and J is the target image. The target image J is obtained by applying a deformation field Φ to the reference image I , as depicted in Fig. 2. This deformation field Φ is described by a transformation function $g(\mathbf{x}; \boldsymbol{\mu}) : V_J \rightarrow V_I$, where V_J and V_I are continuous domains on which J and I are defined, and $\boldsymbol{\mu}$ is a set of transformation parameters to be determined.

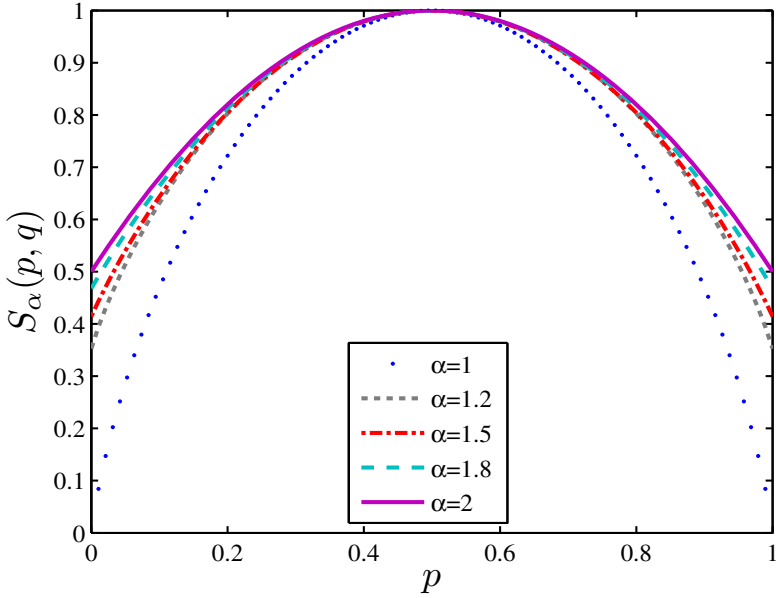


Fig. 1. JT similarity $S_\alpha(\mathbf{p}, \mathbf{q})$ between two Bernoulli distributions $\mathbf{p} = (p, 1 - p)$ and $\mathbf{q} = (1 - p, p)$ for different values of α

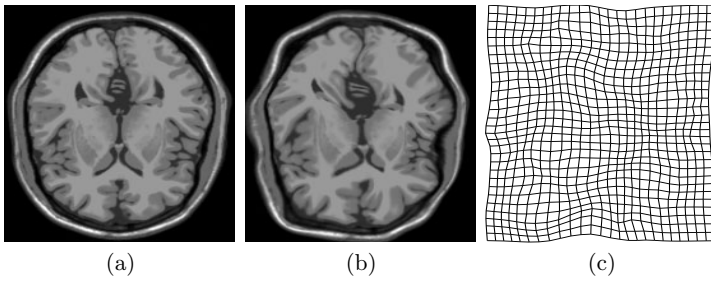


Fig. 2. (a) Reference image I ; (b) target image J ; (c) deformation field Φ

Then, the image registration problem may be formulated as an optimization problem

$$\hat{\boldsymbol{\mu}} = \arg \min_{\boldsymbol{\mu}} S_\alpha^\omega(I(\mathbf{x}), J(g(\mathbf{x}; \boldsymbol{\mu}))). \tag{5}$$

To align the transformed target image $J(g(\mathbf{x}; \boldsymbol{\mu}))$ to the reference image I , we seek the set of transformation parameters $\boldsymbol{\mu}$ that minimize the image cost function $S_\alpha^\omega(I(\mathbf{x}), J(g(\mathbf{x}; \boldsymbol{\mu})))$.

2.3 Transformation Model

Several transformation models have been proposed over the years to represent a nonrigid deformation field. In this paper, we model the transformation $g(\mathbf{x}; \boldsymbol{\mu})$ using the free form deformation [11], which is based on cubic B-splines [18]. Let Φ denote a $n_x \times n_y \times n_z$ mesh of control points $\varphi_{i,j,k}$ with a uniform spacing Δ . Then, the 3D transformation at any point $\mathbf{x} = [x, y, z]^T$ in the target image is interpolated using a linear combination of a cubic B-spline convolution kernel as follows

$$g(\mathbf{x}; \boldsymbol{\mu}) = \sum_{ijk} \eta_{ijk} \beta^{(3)} \left(\frac{\mathbf{x} - \varphi_{ijk}}{\Delta} \right), \quad (6)$$

where $\beta^{(3)}(\mathbf{x}) = \beta^{(3)}(x)\beta^{(3)}(y)\beta^{(3)}(z)$ is a separable cubic B-spline convolution kernel, and η_{ijk} are the deformation coefficients associated to the control points φ_{ijk} . The degree of nonrigidity can be adopted to a specific registration problem by varying the mesh spacing or the resolution of the mesh Φ .

2.4 Optimization of the JT Similarity

We adopt a limited memory quasi-Newton method for solving the optimization problem given by Eq. (5). The calculation of the analytical gradient of the objective (JT similarity) function is necessary to not only avoid discretization errors and also to achieve an efficient and robust optimization scheme.

Denote by $\mathcal{X} = \{x_1, x_2, \dots, x_n\}$ and $\mathcal{Y} = \{y_1, y_2, \dots, y_n\}$ the sets of pixel intensity values of the reference image $I(\mathbf{x})$ and the deformed target image $J(g(\mathbf{x}; \boldsymbol{\mu}))$, respectively. Let X and Y be two random variables taking values in \mathcal{X} and \mathcal{Y} . Then, we define the conditional intensity probability distributions \mathbf{p}_i as

$$\mathbf{p}_i = \mathbf{p}_i(J(g(\mathbf{x}; \boldsymbol{\mu})|I(\mathbf{x})) = (p_{ij})_{j=1, \dots, n}, \quad \forall i = 1, \dots, n,$$

where $p_{ij} = P(Y = y_j | X = x_i) = p(j|i; \boldsymbol{\mu})$, $j = 1, \dots, n$. Note that in p_{ij} the parameter vector $\boldsymbol{\mu}$ is omitted for notational simplicity. It is worth pointing out that if the images I and J are exactly matched, then $\mathbf{p}_i = (\delta_{ij})$ and by Proposition 1, the JT divergence is therefore maximized and consequently the JT similarity measure is minimized.

2.5 Derivative of the JT Similarity

In a typical registration problem, direct access to the marginal and joint probability densities is not available and hence the densities must be estimated from the image data. Parzen windows can be used for this purpose. That is, the densities are constructed by taking intensity samples from the image and superpositioning kernel functions centered on the elements of these samples. We propose to use the B-spline Parzen window to estimate the conditional intensity probability of the interpolated target image given the reference image. Let $\beta^{(0)}$ be a zero-order spline Parzen window and $\beta^{(3)}$ be a cubic spline Parzen window,

then the smoothed conditional probability of $J(g(\mathbf{x}; \boldsymbol{\mu}))$ given $I(\mathbf{x})$ is expressed as:

$$p(j|i; \boldsymbol{\mu}) = \frac{p(j, i; \boldsymbol{\mu})}{p_I(i)}, \tag{7}$$

where

$$p(j, i; \boldsymbol{\mu}) = \xi \sum_{\mathbf{x} \in V} \beta^{(0)} \left(i - \frac{I(\mathbf{x}) - f_I^0}{\Delta b_I} \right) \beta^{(3)} \left(j - \frac{J(g(\mathbf{x}; \boldsymbol{\mu})) - f_J^0}{\Delta b_J} \right), \tag{8}$$

and

$$p_I(i) = \xi \sum_{\mathbf{x} \in V} \beta^{(0)} \left(i - \frac{I(\mathbf{x}) - f_I^0}{\Delta b_I} \right). \tag{9}$$

The normalization factor ξ ensures sum to one of the probabilities, and $I(\mathbf{x})$ and $J(g(\mathbf{x}; \boldsymbol{\mu}))$ are samples of the reference and interpolated target images respectively. These samples are normalized by the minimum intensity value, f_I^0, f_J^0 , and intensity range of each bin, Δb_I and Δb_J respectively. The marginal probability $p_I(i)$ is independent of the transformation parameters, and does not contribute to the cost function derivative. Thus, it does not need to be smooth. Hence, a zero order B-spline kernel is used. By taking the derivative of the conditional probability with respect to $\boldsymbol{\mu}$, we get

$$\begin{aligned} \frac{\partial p(j|i; \boldsymbol{\mu})}{\partial \boldsymbol{\mu}} &= \frac{\gamma}{p_I(i)\Delta b_J} \sum_{\mathbf{x} \in V} \beta^{(0)} \left(i - \frac{I(\mathbf{x}) - f_I^0}{\Delta b_I} \right) \beta^{(3)} \left(j - \frac{J(g(\mathbf{x}; \boldsymbol{\mu})) - f_J^0}{\Delta b_J} \right) \\ &\quad \cdot \left(\frac{\partial J(t)}{\partial t} \Big|_{t=g(\mathbf{x}; \boldsymbol{\mu})} \right) \frac{\partial g(\mathbf{x}; \boldsymbol{\mu})}{\partial \boldsymbol{\mu}}, \end{aligned} \tag{10}$$

where $\beta^{(3)}$ is the derivative of the cubic spline kernel:

$$\beta^{(3)}(u) = \begin{cases} 0.0 & u \leq -2 \\ 2u + 2 + \frac{1}{2}u^2 & -2 < u \leq -1 \\ -2u - \frac{3}{2}u^2 & -1 < u \leq 0 \\ -2u + \frac{3}{2}u^2 & 0 < u \leq 1 \\ 2u - 2 - \frac{1}{2}u^2 & 1 < u \leq 2 \\ 0.0 & u > 2 \end{cases} \tag{11}$$

The JT similarity and its derivative are given by

$$\begin{cases} S_\alpha^\omega(\mathbf{p}_1, \dots, \mathbf{p}_n) = 1 - \frac{D_\alpha^\omega(\mathbf{p}_1, \dots, \mathbf{p}_n)}{\log_\alpha n} \\ \frac{\partial S_\alpha^\omega(\mathbf{p}_1, \dots, \mathbf{p}_n)}{\partial \boldsymbol{\mu}} = -\frac{\partial D_\alpha^\omega(\mathbf{p}_1, \dots, \mathbf{p}_n)}{\partial \boldsymbol{\mu}} \times \frac{1}{\log_\alpha n} \end{cases} \tag{12}$$

2.6 Summary of the Proposed Algorithm

The proposed algorithm is implemented by changing the deformation in the target image until the discrepancy between the target and reference images is minimized. The main algorithmic steps of our nonrigid image registration approach are summarized in Algorithm 1. First, the algorithm initializes the deformation field Φ by creating a uniform B-spline control grid with predefined spacing knots. Next, a 3-level hierarchical multi-resolution scheme is used to achieve the best compromise between the registration accuracy and the associated computational cost. As the hierarchical level increases the resolution of the control mesh is increased, along with the image resolution, in a coarse to fine fashion. In each hierarchical level, a limited-memory, quasi-Newton minimization scheme is used to find the optimum set of transformation parameters that reduce the JT cost function until the difference between the cost function values in two consecutive iterations is less than $\epsilon = 0.01$. The resolution of the optimum set of transformation parameters, at a coarser level, is increased to be used as starting point for the next hierarchical level.

Algorithm 1. Proposed nonrigid registration approach

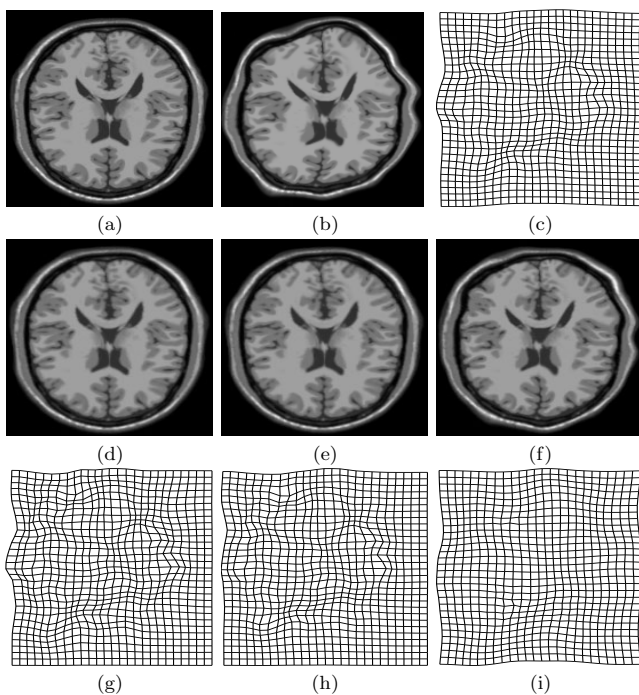
- 1: Initialize the deformation field Φ
 - 2: **for** *hierarchicalLevel* = 1 to 3 **do**
 - 3: Calculate the cost function and its gradient as given by Eq. (12)
 - 4: **repeat**
 - 5: Use the quasi-Newton method to solve the optimization problem given by Eq. (5)
 - 6: Update the deformation field
 - 7: Recalculate the cost function and its gradient
 - 8: **until** the difference in consecutive iterates is less than $\epsilon = 0.01$
 - 9: Increase the resolution of both the deformation field and the image.
 - 10: **end for**
-

3 Experimental Results

We tested the performance of the proposed approach on a medical imaging dataset that was obtained from the brainweb database at the Montreal Neurological Institute [20]. This dataset contains a full 3D simulated brain MR data volumes from several protocols, including T1-weighted (MR-T1), T2-weighted (MR-T2), and proton density (MR-PD) with a variety of slice thicknesses, noise levels, and levels of intensity non-uniformity. All the corresponding slices from different protocols are originally aligned with each other. The images used in our experiments have $181 \times 217 \times 181$ voxels with a 1 mm voxel size in each dimension. To validate the nonrigid registration accuracy of the proposed method, we first applied both geometric and intensity distortions to the reference image in order to generate the target image. Then, we aligned the target image with the reference image. We also compared the image registration results of our

Table 1. Mean square error (MSE) statistics of the estimated nonrigid deformation

μ_g	our method		RC		NMI	
	$\hat{\mu}_{\text{MSE}}$	$\hat{\sigma}_{\text{MSE}}$	$\hat{\mu}_{\text{MSE}}$	$\hat{\sigma}_{\text{MSE}}$	$\hat{\mu}_{\text{MSE}}$	$\hat{\sigma}_{\text{MSE}}$
2.1	0.4982	0.0325	0.5013	0.0251	0.7133	0.0436
3.0	0.5971	0.0843	0.5994	0.0413	0.8936	0.0362
4.6	0.7002	0.0157	0.7342	0.0321	1.0933	0.0733
5.1	0.7621	0.0241	0.7998	0.0379	1.4992	0.0536
5.7	0.7922	0.0252	0.8213	0.0317	1.6916	0.0837

**Fig. 3.** Geometric distortion experiment : (a) MR-T1 image; (b) distorted MR-T1 image with geometric distortion; (c) ground truth deformation field; (d)-(f) registered images using our approach, RC, and NMI, respectively; (g)-(i) estimated deformation fields using our approach, RC, and NMI, respectively

approach to RC and NMI approaches, which are implemented in the Medical Image Registration Toolbox (MIRT) [13] and in the Image Registration Toolkit (ITK) [14], respectively. In all the experiments we used an entropic index $\alpha = 2$ and the normalized histogram of the reference image as the weight vector ω for the JT similarity measure. In the first experiment, we distorted the reference image MR-T1 with a known nonrigid transformation field, or the so-called

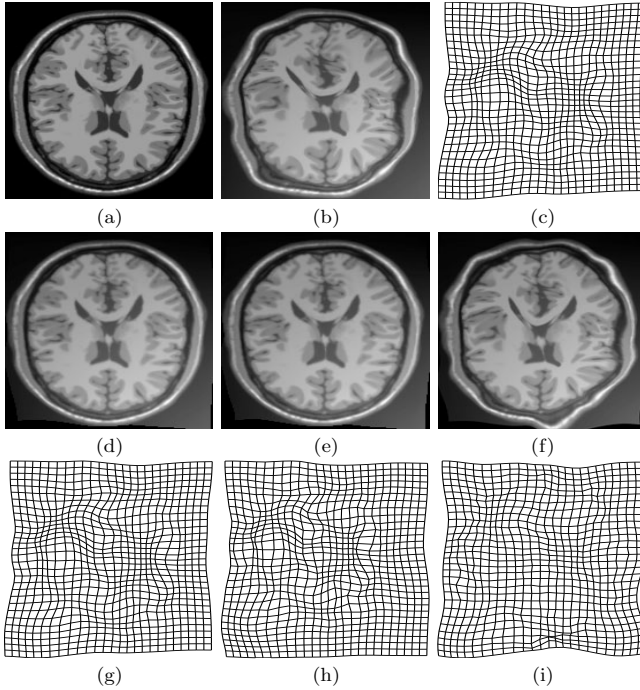


Fig. 4. Geometric and intensity distortion experiment : (a) MR-T1 image; (b) distorted MR-T1 image with geometric and intensity distortion; (c) ground truth deformation field; (d)-(f) registered images using our approach, RC, and NMI, respectively; (g)-(i) estimated deformation fields using our approach, RC, and NMI, respectively

ground truth deformation. Then, we applied the proposed registration approach, RC and NMI algorithms. And finally, we compared the obtained deformations fields with the ground truth. Fig. 3 depicts the results obtained from this experiment. Note that the registered target images obtained using the proposed approach and RC approach are visually more similar in shape to the target image than the image produced by the NMI approach. Moreover, the estimated transformation field resulted from our algorithm is more similar to the ground truth than of the field deformations obtained using RC and NMI approaches. To measure the registration accuracy of the proposed method, we computed the mean ($\hat{\mu}_{\text{MSE}}$) and standard deviation ($\hat{\sigma}_{\text{MSE}}$) of the mean squared error (MSE) between the ground truth and estimated displacement vectors. Table 1 displays the MSE statistics of the estimated nonrigid deformation, when compared to the ground truth. The first column shows the mean ground truth deformation, which represents the magnitude of the displacement vector that is used to generate the target images in each experiment. For each row twenty different transformation fields with this mean are generated and applied to the reference image in order to

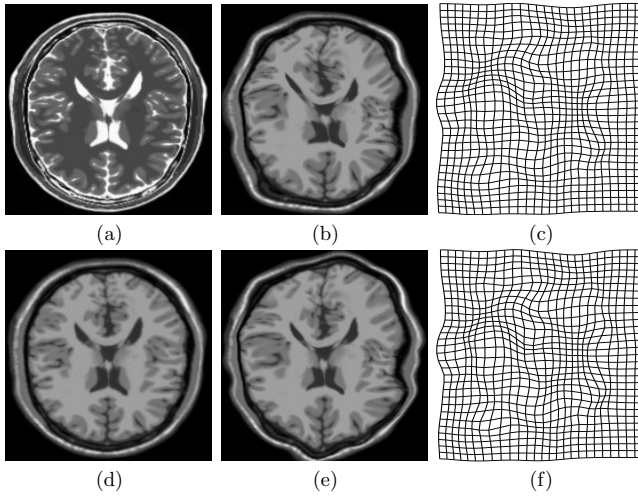


Fig. 5. Multimodality experiment: (a) MR-T2 image; (b) distorted MR-T1 image with geometric distortion; (c) ground truth deformation field; (d) and (e) registered images using our approach, and NMI, respectively; (f) the estimated transformation using our approach

generate the corresponding target images. The second and third columns display the average and standard deviation of MSE for the generated twenty pairs of reference-target images. The results obtained using the proposed approach are considerably small compared to those of RC and NMI methods.

In the second experiment, we used similar steps as in the first experiment, but this time we generated the target image by distorting the reference image with both intensity and geometric distortions. The intensity distortion is generated by corrupting the reference image as follows [13]:

$$\begin{aligned}
 & - I(x, y) = I^\gamma(x, y) + v \frac{xy}{MN} + \frac{1}{K} \sum_{k=1}^K \exp\left(-\frac{\| [x; y] - \Psi_K \|^2}{2\sigma^2}\right) \\
 & - \text{Rescale } I \text{ to } [0, 1],
 \end{aligned}$$

where the first term represents the gamma correction on I after geometric distortion, the second term models a smoothly varying global intensity field and the third term models locally-varying intensity field with a mixture of K Gaussian densities. In this experiment, we chose a distortion level 2 with parameters as follows: $v = 0.4, K = 1, \Psi_K$ were randomly selected from the interval $[1, \nu]$ (ν is the size of the image domain), $\sigma = 30$, and γ is selected randomly from $[0.9, 1.2]$. The registered images shown in Fig. 4 demonstrate that the proposed algorithm outperforms the RC approach, in the presence of spatially-varying intensity distortion. The result obtained by the NMI approach shows a poor performance. Moreover, the estimated deformation field obtained by our approach shows superior accuracy in comparison to RC and NMI methods.

3.1 Multimodality Test

The images used in this experiment are corresponding slices from MR-T1 and MR-T2 image pair, and they are originally aligned with each other. In this experiment we registered the geometrically deformed MR-T1 image onto MR-T2 image using our approach and the NMI method. We omitted the result of the RC approach because it is not applicable to multimodal images. Fig. 5 shows the accuracy of our method in registering images from different modalities. As can be seen, the registered image using the NMI approach still has a considerable amount of misregistration. However, most of the visible amount of misalignment in the target image has been removed after applying the proposed approach. In addition, the nonrigid transformation estimated by the proposed method looks very similar to the ground truth, indicating a much better performance of our approach.

4 Conclusions

An information-theoretic framework for nonrigid image registration was proposed. The experimental results on a medical imaging dataset indicate the feasibility of the proposed approach and a much better performance compared to RC and NMI methods in terms of registration accuracy in the presence of intensity and geometric distortions. Future work will focus on extending our approach to nonrigid multimodal image registration.

Acknowledgment

This work was supported in part by NSERC discovery grant.

References

1. Andronache, A., Siebenthal, M., Szekely, G., Cattin, P.: Non-rigid registration of multi-modal images using both mutual information and cross-correlation. *Medical Image Analysis* 12, 3–15 (2008)
2. Bro-Nielsen, M., Gramkow, C.: Fast fluid registration of medical images. In: *Proc. Int. Conf. on Visualization in Biomedical Computing*, pp. 267–276 (1993)
3. Christensen, G., Rabbitt, R., Miller, M.: Deformable templates using large deformation kinematics. *IEEE Trans. on Image Processing* 5, 1435–1447 (1996)
4. Davatzikos, C.: Spatial transformation and registration of brain images using elastically deformable models. *Computer Vision and Image Understanding* 66, 207–222 (1997)
5. Gee, J., Reivich, K., Bajcsy, R.: Elastically deforming 3D atlas to match anatomical brain images. *Journal of Computer Assisted Tomography* 17, 225–236 (1993)
6. Hajnal, J.V., Hill, D., Hawkes, D.J.: *Medical Image Registration*. CRC Press, Boca Raton (2001)
7. Likar, B., Pernus, F.: A hierarchical approach to elastic registration based on mutual information. *Image and Vision Computing* 19, 33–44 (2001)

8. Loeckx, D., Slagmolen, P., Maes, F., Vandermeulen, D., Suetens, P.: Nonrigid image registration using conditional mutual information. *IEEE Trans. on Medical Imaging* 29, 19–29 (2007)
9. Maes, F., Collignon, A., Vandermeulen, D., Marchal, G., Suetens, P.: Multimodality image registration by maximization of mutual information. *IEEE Trans. on Medical Imaging* 16, 187–198 (1997)
10. Maes, F., Vandermeulen, D., Suetens, P.: Medical image registration using mutual information. *Proceedings of the IEEE* 91, 1699–1722 (2003)
11. Mattes, D., Haynor, D.R., Vesselle, H., Lewellen, T.K., Eubank, W.: PET-CT image registration in the chest using free-form deformations. *IEEE Trans. on Medical Imaging* 22, 120–128 (2003)
12. Mohamed, W., Zhang, Y., Ben Hamza, A., Bouguila, N.: Stochastic optimization approach for entropic image alignment. In: *Proc. IEEE Int. Symposium on Information Theory*, pp. 2126–2130 (2008)
13. Myronenko, A., Song, X.: Image registration by minimization of residual complexity. In: *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 49–56 (2009)
14. Rueckert, D., Sonoda, L., Hayes, C., Hill, D., Leach, M., Hawkes, D.J.: Nonrigid registration using free-form deformations: Application to breast MR images. *IEEE Trans. on Medical Imaging* 18, 712–721 (1999)
15. Studholme, C., Hill, D., Hawkes, D.J.: An overlap invariant entropy measure of 3D medical image alignment. *Pattern Recognition* 32, 71–86 (1999)
16. Tsallis, C.: Possible generalization of Boltzmann-Gibbs statistics. *Journal of Statistical Physics* 52, 479–487 (1988)
17. Viola, P., Wells, W.M.: Alignment by maximization of mutual information. *International Journal of Computer Vision* 24, 154–173 (1997)
18. Wahba, G.: *Spline models for observational data*. SIAM, Philadelphia (1990)
19. Wang, F., Vemuri, B.: Non-rigid multi-modal image registration using cross-cumulative residual entropy. *International Journal of Computer Vision* 74, 201–215 (2007)
20. [Online] <http://www.bic.mni.mcgill.ca/brainweb/>

Precipitates Segmentation from Scanning Electron Microscope Images through Machine Learning Techniques

João P. Papa¹, Clayton R. Pereira¹, Victor H.C. de Albuquerque², Cleiton C. Silva³, Alexandre X. Falcão⁴, and João Manuel R.S. Tavares⁵

¹ Dep. of Computing, UNESP - Univ Estadual Paulista, Bauru, Brazil
`{papa,clayton}@fc.unesp.br`

² Center of Technological Sciences, University of Fortaleza, Fortaleza, Brazil
`victor.albuquerque@fe.up.pt`

³ Dep. of Materials and Metallurgical Engineering,
Federal University of Ceará, Brazil
`cleitonufc@yahoo.com.br`

⁴ Institute of Computing, State University of Campinas, Campinas, Brazil
`afalcao@ic.unicamp.br`

⁵ Faculty of Engineering, University of Porto, Porto, Portugal
`tavares@fe.up.pt`

Abstract. The presence of precipitates in metallic materials affects its durability, resistance and mechanical properties. Hence, its automatic identification by image processing and machine learning techniques may lead to reliable and efficient assessments on the materials. In this paper, we introduce four widely used supervised pattern recognition techniques to accomplish metallic precipitates segmentation in scanning electron microscope images from dissimilar welding on a Hastelloy C-276 alloy: Support Vector Machines, Optimum-Path Forest, Self Organizing Maps and a Bayesian classifier. Experimental results demonstrated that all classifiers achieved similar recognition rates with good results validated by an expert in metallographic image analysis.

Keywords: Support Vector Machines, Optimum-Path Forest, Scanning Electron Microscope, Metallic Precipitates Segmentation, Hastelloy C-276.

1 Introduction

Nickel based alloys are an important class of metallic materials especially employed under severe operational conditions, mainly because of its high temperature strength and resistance to corrosion/oxidation. In this context, Hastelloy C276 alloy has been notably used as protective coating against corrosion on inner surface in equipments from petroleum and petrochemical industries due to the high contents of chromium, molybdenum, and tungsten.

Among many manufacturing process used to deposit the coating, the arc welding process is one of the most important. During the solidification of the liquid metal in the weld pool, a phenomenon of microsegregation from solid dendrite to liquid interdendritic is observed for some elements as molybdenum and tungsten [14].

In the final stage of solidification, the liquid enriched in Mo (molybdenum) and W (tungsten) originates a new phase, known as topologically closed packed (TCP) phase [14], which is detrimental to mechanical properties due to its hard and brittle nature [1]. Besides that, the resistance to corrosion of these type of alloys can be decreased by the precipitation of the Mo-rich TCP phase. In addition, the formation of TCP phases are responsible for the weld metal hot cracks in Hastelloy C-276 [5]. In order to avoid or minimize these deleterious phases, it has a consensus about choosing the welding parameters in a properly manner.

Nonetheless, to verify the effect of welding parameters on the formation of TCP phases it is often necessary to accurately identify the amount of precipitates, which are responsible to decrease the mechanical properties of metallic materials. Thus, it is very important to have an effective tool to identify and further quantity the material precipitates and microstructures, in order to assess the quality of metallic materials as a whole.

Albuquerque et al. [2] have addressed this problem using image processing techniques together with machine learning ones in order to speed up the process and to make it less prone to errors inherent to human inspection. In that work, it was presented and evaluated computational solutions for segmentation and quantification of different types of cast iron microstructures from optical microscopy images based on Artificial Neuronal Networks with Multilayer Perceptron (ANN-MLP) and Self-Organizing Maps (SOM), which were compared against a commercial system. As far as we know, only Albuquerque et al. [4] tackled the problem of material precipitates segmentation using images obtained from scanning electron microscope (SEM).

Hence, we propose in this work to apply machine learning techniques that have not been applied to this context up to date, such as: Optimum-Path Forest (OPF), two different implementations of Support Vector Machines (SVMs), SOM and a Bayesian classifier. The remainder of the paper is organized as follows. As the OPF classifier was recently introduced, we dedicated a Section to introduce its fundamentals and learning algorithm (Section 2). Section 3 addresses the methodology and the used dataset. The experimental results are addressed in Section 4 and the conclusions are stated in Section 5.

2 Pattern Recognition by Optimum-Path Forest

The Optimum-Path Forest is a framework to assist the development of pattern recognition techniques based on optimum-path forest [12]. An OPF-based classifier models the problem of pattern recognition as a graph partition in a feature space induced by the dataset. Each sample is represented by a set of features

and a distance function measures their dissimilarity in the feature space. The training samples are then interpreted as the nodes of a graph, whose arcs are defined by a given adjacency relation and weighted by the distance function. It is expected that samples from a same class/cluster are connected by a path of nearby samples. Therefore, the degree of connectedness for any given path is measured by a connectivity (path-value) function, which exploits the distances along the path.

In supervised learning, the true label of the training samples is known and so it is exploited to identify key samples (prototypes) in each class. Optimum paths are computed from the prototypes to each training sample, such that each prototype becomes root of an optimum-path tree composed by its most strongly connected samples. The labels of these samples are assumed to be the same of their root. In unsupervised learning, each cluster is represented by an optimum-path tree rooted at a single prototype but we do not know the class label of the training samples. Therefore, we expect that each cluster contains only samples of a same class and some other information about the application is needed to complete classification.

The basic idea is then to specify an adjacency relation and a path-value function, compute prototypes and reduce the problem into an optimum-path forest computation in the underlying graph. The training forest becomes a classifier which can assign to any new sample the label of its most strongly connected root. Essentially, this methodology extends a previous approach, called *Image Foresting Transform* [8], for the design of image processing operators from the image domain to the feature space.

Papa et al. [11] presented a first method for supervised classification using a complete graph (implicit representation) and the maximum arc weight along a path as connectivity function. The prototypes were chosen as samples that share an arc between distinct classes in a minimum spanning tree of the training set [7].

Another supervised learning method was proposed in [10]. In this case, the arcs connect k -nearest neighbors (k -nn) in the feature space. The distances between adjacent nodes are used to estimate a probability density value of each node and optimum paths are computed from the maxima of this probability density function (pdf). For large datasets, we usually use a smaller training set and a much larger evaluation set to learn the most representative samples from the classification errors in the evaluation set. This considerably improves classification accuracy of new samples. This strategy was assessed with k -nn graphs in [13]. The accuracy results can be better than using similar strategy with complete graph [11] for some situations, but the latter is still preferred because it is faster and does not require the optimization of the parameter k .

An unsupervised version of OPF was presented by Rocha et al. [15], which is quite similar to the supervised one with k -nn graph. The main difference rely on the estimation of the best k value: in this case, as we do not have information about labels, the k value chosen is the one that minimizes the minimum cut over the whole graph. In this paper, we adopted the OPF with complete graph,

since that this version is the most used. For sake of simplicity, any further reference to OPF will mean this version. The next sections will details the training, classification and the learning with pruning procedures for OPF.

2.1 Training

In large datasets, the number of labeled samples for training is usually large. Therefore, a first strategy to make a classifier more efficient is the use of two labeled and disjoint sets, Z_1 and Z_2 , $|Z_1| \ll |Z_2|$, being the first the actual training set and the second an evaluation set. The purpose of the evaluation set is to improve the quality of the samples in the training set, without increasing its size, by replacing classification errors in Z_2 by non-prototype samples of Z_1 [11]. After this learning process, the classifier is ready to be tested on any unseen dataset Z_3 . For validation purpose, this process must also be repeated several times, with different random and disjoint sets Z_1 , Z_2 , and Z_3 , in order to obtain the average accuracy results.

Let (Z_1, A) be a complete graph whose nodes are the samples in Z_1 and any pair of samples defines an arc in $A = Z_1 \times Z_1$. The arcs do not need to be stored and so the graph representation is implicit. A path is a sequence of distinct samples $\pi_t = \langle s_1, s_2, \dots, s_{k-1}, t \rangle$ with terminus t , where $(s_i, s_{i+1}) \in A$ for $1 \leq i \leq k - 1$. A path is said *trivial* if $\pi_t = \langle t \rangle$. We assign to each path π_t a cost $f(\pi_t)$ given by a path-value function f . A path π_t is considered optimum if $f(\pi_t) \leq f(\tau_t)$ for any other path τ_t with the same terminus t . We also denote by $\pi_s \cdot \langle s, t \rangle$ the concatenation of a path π_s and arc (s, t) .

Training essentially consists of finding an optimum-path forest in (Z_1, A) , which is rooted in a special set $S \subset Z_1$ of prototypes. As proposed in [11], the set S is represented by samples that share arcs between distinct classes in a minimum-spanning tree (MST) of (Z_1, A) [7]. For path-value function f_{max} , these prototypes (roots of the forest) tend to minimize the classification errors in Z_1 , when their labels are propagated to the nodes of their trees:

$$\begin{aligned}
 f_{max}(\langle s \rangle) &= \begin{cases} 0 & \text{if } s \in S \\ +\infty & \text{otherwise,} \end{cases} \\
 f_{max}(\pi_s \cdot \langle s, t \rangle) &= \max\{f_{max}(\pi_s), d(s, t)\},
 \end{aligned} \tag{1}$$

such that $f_{max}(\pi_s)$ computes the maximum distance between adjacent samples in a non-trivial path π_s .

The training algorithm for the OPF classifier [11] assigns one optimum path $P_1^*(s)$ from S to every sample $s \in Z_1$, forming an optimum path forest P_1 (a function with no cycles which assigns to each $s \in Z_1 \setminus S$ its predecessor $P_1(s)$ in $P_1^*(s)$ or a marker *nil* when $s \in S$). Let $R_1(s) \in S$ be the root of $P_1^*(s)$ (which can be reached from $P_1(s)$), the OPF algorithm computes for each $s \in Z_1$, the minimum cost $C_1(s)$ of $P_1^*(s)$, the class label $L_1(s) = \lambda(R_1(s))$, and the predecessor $P_1(s)$. *Algorithm 7* implements this training procedure.

Algorithm 1 – TRAINING ALGORITHM

INPUT: A λ -labeled training set Z_1 and the pair (v, d) for feature vector and distance computations.

OUTPUT: Optimum-path forest P_1 , cost map C_1 , label map L_1 , and ordered set Z'_1 .

AUXILIARY: Priority queue Q , set S of prototypes, and cost variable cst .

1. Set $Z'_1 \leftarrow \emptyset$ and compute by MST the prototype set $S \subset Z_1$.
2. For each $s \in Z_1 \setminus S$, set $C_1(s) \leftarrow +\infty$.
3. For each $s \in S$, do
4. $C_1(s) \leftarrow 0$, $P_1(s) \leftarrow nil$, $L_1(s) \leftarrow \lambda(s)$, and insert s in Q .
5. While Q is not empty, do
6. Remove from Q a sample s such that $C_1(s)$ is minimum.
7. Insert s in Z'_1 .
8. For each $t \in Z_1$ such that $t \neq s$ and $C_1(t) > C_1(s)$, do
9. Compute $cst \leftarrow \max\{C_1(s), d(s, t)\}$.
10. If $cst < C_1(t)$, then
11. If $C_1(t) \neq +\infty$, then remove t from Q .
12. $P_1(t) \leftarrow s$, $L_1(t) \leftarrow L_1(s)$, $C_1(t) \leftarrow cst$.
13. Insert t in Q .
14. Return a classifier $[P_1, C_1, L_1, Z'_1]$.

2.2 Classification

In [11], the classification of each new sample $t \in Z_2$ (or Z_3) is done based on the distance $d(s, t)$ between t and each training node $s \in Z_1$ and on the evaluation of the following equation:

$$C_2(t) = \min\{\max\{C_1(s), d(s, t)\}\}, \forall s \in Z_1. \tag{2}$$

Let $s^* \in Z'_1$ be the node s that satisfies Equation (2). It essentially considers all possible paths π_s from S in (Z_1, A) extended to t by an arc (s, t) , finds the optimum path $P_1^*(s^*) \cdot \langle s^*, t \rangle$, and label t with the class $\lambda(R_1(s^*))$ of its most strongly connected prototype $R_1(s^*) \in S$ (i.e., $L_2(t) \leftarrow L_1(s^*) = \lambda(R_1(s^*))$).

Note that Z_1 can be replaced by Z'_1 in Equation (2) and its evaluation can halt when $\max\{C_1(s), d(s, t)\} < C_1(s')$ for a node s' whose position in Z'_1 succeeds the position of s . This avoids to visit all nodes in Z'_1 in many cases and the efficiency gain increases with the time complexity of $d(s, t)$. Algorithm 2 implements this scheme for classification procedure.

In Algorithm 2, the main loop (Lines 1 – 9) performs classification of all nodes in Z_2 . The inner loop (Lines 4 – 9) visits each node $k_{i+1} \in Z'_1$, $i = 1, 2, \dots, |Z'_1|$ until an optimum path $\pi_{k_{i+1}} \cdot \langle k_{i+1}, t \rangle$ be found. In the worst scenario, it visits all nodes in Z'_1 (Line 4). Line 5 evaluates $f_{max}(\pi_{k_{i+1}} \cdot \langle k_{i+1}, t \rangle)$ and Lines 7 – 8 updates cost, label and predecessor of t whenever $\pi_{k_{i+1}} \cdot \langle k_{i+1}, t \rangle$ is better than the current path π_t (Line 6).

Algorithm 2 – OPF CLASSIFICATION

INPUT: Classifier $[P_1, C_1, L_1, Z'_1]$, evaluation set Z_2 (or test set Z_3), and the pair (v, d) for feature vector and distance computations.
 OUTPUT: Label L_2 and predecessor P_2 maps defined for Z_2 .
 AUXILIARY: Cost variables tmp and $mincost$.

1. For each $t \in Z_2$, do
2. $i \leftarrow 1, mincost \leftarrow \max\{C_1(k_i), d(k_i, t)\}.$
3. $L_2(t) \leftarrow L_1(k_i)$ and $P_2(t) \leftarrow k_i.$
4. While $i < |Z'_1|$ and $mincost > C_1(k_{i+1}),$ do
5. Compute $tmp \leftarrow \max\{C_1(k_{i+1}, d(k_{i+1}, t))\}.$
6. If $tmp < mincost,$ then
7. $mincost \leftarrow tmp.$
8. $L_2(t) \leftarrow L(k_{i+1})$ and $P_2(t) \leftarrow k_{i+1}.$
9. $i \leftarrow i + 1.$
10. Return $[L_2, P_2].$

2.3 Pruning Irrelevant Patterns

Large datasets usually present redundancy, so at least in theory it should be possible to estimate a reduced training set with the most relevant patterns for classification. The use of a training set Z_1 and an evaluation set Z_2 has allowed us to learn relevant samples for Z_1 from the classification errors in Z_2 , by swapping misclassified samples of Z_2 and non-prototype samples of Z_1 during a few iterations [11]. In this learning strategy, Z_1 remains with the same size and the classifier instance with the highest accuracy is selected to be tested in the unseen set Z_3 . In this section, we use this learning procedure (as described in Algorithm 3) within a method (Algorithm 4) to reduce the training set size by identifying and eliminating irrelevant samples from Z_1 .

Algorithm 3 – OPF LEARNING ALGORITHM

INPUT: A λ -labeled training and evaluating sets Z_1 and Z_2 , respectively, number T of iterations, and the pair (v, d) for feature vector and distance computations.
 OUTPUT: Optimum-path forest P_1 , cost map C_1 , label map L_1 , and ordered set Z'_1 .
 AUXILIARY: Arrays FP and FN of sizes c for false positives and false negatives, set S of prototypes, and list LM of misclassified samples.

1. Set $MaxAcc \leftarrow -1.$
2. For each iteration $I = 1, 2, \dots, T,$ do
3. $LM \leftarrow \emptyset$ and compute the set $S \subset Z_1$ of prototypes.
4. $[P_1, C_1, L_1, Z'_1] \leftarrow$ Algorithm 1 ($Z_1, S, (v, d)$).
5. For each class $i = 1, 2, \dots, c,$ do
6. $FP(i) \leftarrow 0$ and $FN(i) \leftarrow 0.$
7. $[L_2, P_2] \leftarrow$ Algorithm 2 ($Z'_1, Z_2, (v, d)$)
8. For each sample $t \in Z_2,$ do
9. If $L_2(t) \neq \lambda(t),$ then

```

10.   |   |   |   FP(L2(t)) ← FP(L2(t)) + 1.
11.   |   |   |   FN(λ(t)) ← FN(λ(t)) + 1.
12.   |   |   |   LM ← LM ∪ t.
13.   |   |   |   Compute accuracy Acc according to [11].
14.   |   |   |   If Acc > MaxAcc then save the current instance [P1, C1, L1, Z'1]
15.   |   |   |   of the classifier and set MaxAcc ← Acc.
16.   |   |   |   While LM ≠ ∅
17.   |   |   |   |   LM ← LM \ t.
18.   |   |   |   |   Replace t by a non-prototype sample, randomly selected from Z1.
19.   |   |   |   Return the classifier instance [P1, C1, L1, Z'1] with the highest accuracy in Z2.

```

The efficacy of *Algorithm 3* increases with the size of Z_1 , because more non-prototype samples can be swapped by misclassified samples of Z_2 . However, for sake of efficiency, we need to choose some reasonable maximum size for Z_1 . After learning the best training samples for Z_1 , we may also mark paths in P_1 used to classify samples in Z_2 and define their nodes as *relevant samples* in a set \mathcal{R} . The “irrelevant” training samples in $Z_1 \setminus \mathcal{R}$ can then be moved to Z_2 . *Algorithm 4* applies this idea repetitively, while the loss in accuracy on Z_2 with respect to the highest accuracy obtained by *Algorithm 3* (using the initial training set size) is less or equal to a maximum value $MLoss$ specified by the user.

Algorithm 4 – LEARNING-WITH-PRUNING ALGORITHM

INPUT: Training and evaluation sets, Z_1 and Z_2 , labeled by λ , the pair (v, d) for feature vector and distance computations, maximum loss $MLoss$ in accuracy on Z_2 , and number T of iterations.
 OUTPUT: EOPF classifier $[P_1, C_1, L_1, Z'_1]$ with reduced training set.
 AUXILIARY: Set \mathcal{R} of relevant samples, and variables Acc and tmp .

```

1.  [P1, C1, L1, Z'1] ← Algorithm 3 (Z1, Z2, T, (v, d)).
2.  [L2, P2] ← Algorithm 2 (Z'1, Z2, (v, d)) and store accuracy in Acc.
3.  tmp ← Acc and R ← ∅.
4.  While |Acc – tmp| ≤ MLoss and R ≠ Z1 do
5.      |   R ← ∅.
6.      |   For each sample t ∈ Z2, do
7.          |   |   s ← P2(t) ∈ Z1.
8.          |   |   While s ≠ nil, do
9.              |   |   |   R ← R ∪ s.
10.             |   |   |   s ← P1(s).
11.             |   |   Move samples from Z1 \ R to Z2.
12.             |   |   [P1, C1, L1, Z'1] ← Algorithm 3 (Z1, Z2, T, (v, d)).
13.             |   |   [L2, P2] ← Algorithm 2 (Z'1, Z2, (v, d)) and store accuracy in tmp.
14.  Return [P1, C1, L1, Z'1].

```

In *Algorithm 4*, Lines 1 – 3 compute learning and classification using the highest accuracy classifier obtained for an initial training set size. Its accuracy is stored in Acc and used as reference value Acc in order to stop the pruning process, when the loss in accuracy is greater than an user-specified value $MLoss$ or all training samples are considered relevant. The main loop in Lines 4 – 13

essentially marks the relevant samples in Z_1 by following backwards the optimum paths used for classification (Lines 5 – 10), moves irrelevant samples to Z_2 , and repeats learning and classification from a reduced training set until it reaches the above stopping criterion.

3 Methodology

The material used to evaluate classifiers was a dissimilar metal weld of Hastelloy C276 alloy on a C-Mn steel substrate. The testing samples were extracted from the welded plates and subsequently went through a metallographic processing, which consists in sanding, polishing and electrolytic etching. All samples were etched using 10% chromic acid, and a 2V tension applied during 15 seconds.

The SEM images were obtained in secondary electron (SE) mode, which presents an adequate contrast between the precipitates and the matrix, due to the enrichment of the precipitates by elements with higher atomic weight such as Mo and W. After the imaging acquisition process, the images were submitted to the analysis of the machine learning solutions under evaluation.

Regarding machine learning techniques, we used here five implementations: Self Organizing Maps (SOM), Optimum-Path Forest (OPF), SVM without kernel mapping (SVM-nokernel), SVM with RBF (Radial Basis Function) as kernel mapping (SVM-RBF) and a Bayesian classifier. For SOM, we used our own implementation with a 5×5 neuronal lattice and 10 iterations for learning. The OPF implementation we used was the one from LibOPF [12], which is a free library of optimum-path forest-based classifiers. For OPF learning algorithm, we used the pruning procedure described in Section 2.3. Regarding SVM-nokernel we adopted LibLINEAR [9] with parameters optimized by cross-validation, and for SVM-RBF we used SVMTorch [6]. Finally, for Bayesian classifier we used our implementation.

As we are working with supervised pattern recognition techniques, it is necessary to have labeled data for the learning process. Thus, we asked for an expert in metallographic image analysis to label an entire image in two classes: foreground (precipitates) and background (matrix). Figure 1 displays the image used to train the classifiers (a) and its respective labeled image (b), in which the red pixels mean the precipitates. In this work, each pixel is considered as a sample to build the dataset, and the feature vector used as input is composed by the gray value of each pixel.

4 Experimental Results

In this section, we present the experiments realized in order to assess the robustness of the classifiers, which were conducted in two phases: in the former (Section 4.1) we used 1% for training and the remaining 99% for classification. The samples were obtained through the whole image shown in Figure 1. In the latter round of experiments (Section 4.2), we used the same 1% above to train the classifiers and further to label another image of the dataset. We conducted

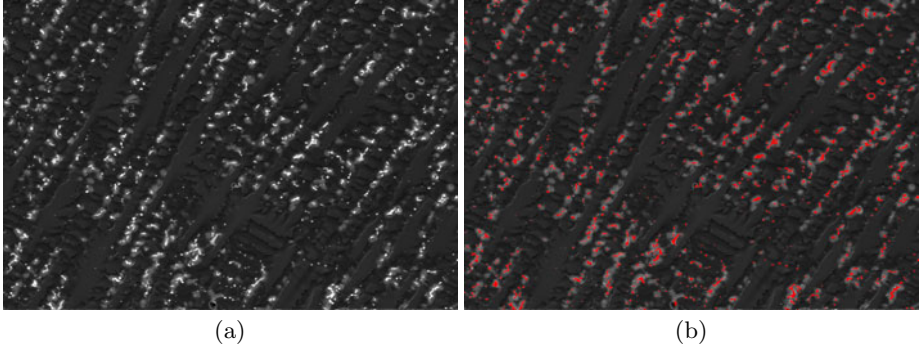


Fig. 1. (a) SEM image used in the first round of experiments (Section 4.1) and (b) after be labeled by an expert

an extra experiment in Section 4.1 in order to assess the performance of OPF learning with pruning algorithm described in Section 2.3. For that experiment, we divided the above 1% in 10% for training and 90% for the evaluating set. The $MLoss$ variable in Algorithm 4 was set to 0.3. Notice that all these values were empirically chosen based on our previous experience.

The accuracies are measured by taking into account that the classes may have different sizes in Z_2 (similar definition is applied for Z_3). If there are two classes, for example, with very different sizes and a classifier always assigns the label of the largest class, its accuracy will fall drastically due to the high error rate on the smallest class.

Let $NZ_2(i)$, $i = 1, 2, \dots, c$, be the number of samples in Z_2 from each class i . We define the errors $e_{i,1}$ and $e_{i,2}$:

$$e_{i,1} = \frac{FP(i)}{|Z_2| - |NZ_2(i)|} \quad \text{and} \quad e_{i,2} = \frac{FN(i)}{|NZ_2(i)|}, \quad i = 1, \dots, c, \quad (3)$$

where $FP(i)$ and $FN(i)$ are the false positives and false negatives, respectively. That is, $FP(i)$ is the number of samples from other classes that were classified as being from the class i in Z_2 , and $FN(i)$ is the number of samples from the class i that were incorrectly classified as being from other classes in Z_2 . The errors $e_{i,1}$ and $e_{i,2}$ are then used to define:

$$E(i) = e_{i,1} + e_{i,2}, \quad (4)$$

where $E(i)$ is the partial sum error of class i . Finally, the accuracy Acc , are defined as:

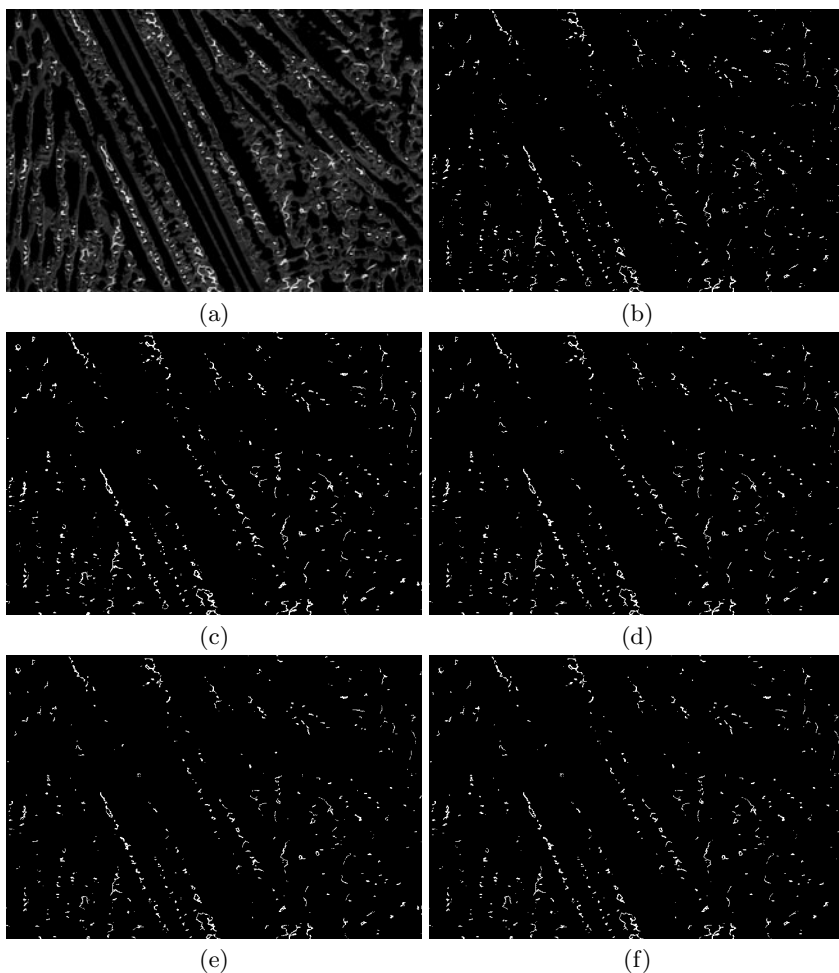
$$Acc = \frac{2c - \sum_{i=1}^c E(i)}{2c} = 1 - \frac{\sum_{i=1}^c E(i)}{2c}. \quad (5)$$

4.1 Robustness of Classifiers

Table 1 display the mean results using 1% for training and 99% for classification after 10 rounds with randomly chosen sets. These experiments were performed

Table 1. Mean accuracy and mean training and classification times for OPF, SVM-RBF, SVM-noKernel and SOM

Classifier	Accuracy %	Training time [s]	Classification Time [s]
OPF	89.86±5.08	0.314	0.594
SVM-RBF	90.84±1.71	0.149	1.672
SVM-noKernel	94.56±2.86	9.965	0.140
SOM	88.87±3.21	0.045	0.065
Bayesian	87.47±1.28	0.025	49.89

**Fig. 2.** (a) SEM image used in the second round of experiments and its respectively classified images by (b) OPF, (c) SVM-noKernel, (d) SVM-RBF, (e) SOM and (f) Bayesian

using a PC with Intel[®] Core I5 processor and 4Gb RAM and Linux Ubuntu 10.04 as the operational system.

Although SVM-noKernel achieved the best results, OPF, SVM-RBF, SOM and Bayesian were also similar if we consider the standard deviation. The fastest classifier for training was Bayesian one, and for classification was SOM, and the best trade-off between efficiency and effectiveness was achieved by OPF, which was the second faster classifier and 11.129 times faster than SVM-noKernel if we take into account the whole execution time, i.e., training plus classification.

Concerning the extra experiment, i.e., the one using OPF learning with pruning algorithm, OPF achieved 91.23% of recognition rate and pruned 97.38% of the training set. Now, the OPF testing time decreased to 0.171 seconds, which turn OPF with training set pruning 3.45 faster than traditional OPF for classification. Note that the accuracy also increased, even reducing the training set.

4.2 Automatic Labeling Images

In this second round of experiments, we applied the same 1% training set used in the previous section to train the classifiers for further labeling another SEM image of the dataset, as illustrated in Figure 2a.

It is possible to observe, from visual assessment, that all approaches achieved reasonable results regarding the quality of the segmentation (Figures 2(b)-(e)). Thus, it feasible to make the Hastelloy C-276 alloy automatic characterization in a precisely and efficiently manner, since that the human inspection may be prone to errors due to the subjectivity of this process, as addressed by Albuquerque et al. [3]. Hence, this work may contribute with a comparison among supervised pattern recognition techniques in order to obtain fast and reliable results to this urged and demanded task.

5 Conclusions

In this paper, we addressed the problem of metallic precipitates segmentation in SEM images, which may affect the material durability and resistance. As there are very few works in this context, the present one assumes a significantly contribution by performing a comparison among the state-of-the-art supervised pattern recognition to accomplish this task.

We conducted two rounds of experiments: (i) in the former, the accuracy of two different implementations of SVMs, SOM and OPF were compared in terms of effectiveness and efficiency for training and classification, and (ii) in the latter experiment we used the classifiers trained in the previous one to label a another image of the dataset. Regarding accuracy over the classification set, all classifiers were similar if we consider the standard deviation, been SOM the fastest one. Additionally, OPF achieved the best trade-off between effectiveness and efficiency. Finally, in the second round, we attested that all classifiers produced similar results to label an image that did not belong to the training set. In addition, these segmentation results were considered feasible by an expert in in metallographic image analysis.

Acknowledgments. The authors thank National Council for Research and Development (CNPq), São Paulo Research Foundation (FAPESP) grants 2009/16206-1 and 2010/02045-3, and Cearense Foundation for the Support of Scientific and Technological Development (FUNCAP) for providing financial support through a DCR grant to UNIFOR for third the author. This work was also partially done in the scope of the project with reference PTDC/EEA-CRO/103320/2008 financially supported by Fundação para a Ciência e a Tecnologia (FCT) in Portugal.

References

1. Akhter, J., Shaikh, M., Ahmad, M., Iqbal, M., Shoaib, K., Ahmad, W.: Effect of aging on the hardness and impact properties of hastelloy c-276. *Journal of Materials Science Letters* 20(4), 333–335 (2001)
2. de Albuquerque, V.H.C., de Alexandria, A.R., Cortez, P.C., Tavares, J.M.R.S.: Evaluation of multilayer perceptron and self-organizing map neural network topologies applied on microstructure segmentation from metallographic images. *NDT & E International* 42(7), 644–651 (2009)
3. de Albuquerque, V.H.C., Cortez, P.C., de Alexandria, A.R., Tavares, J.M.R.S.: A new solution for automatic microstructures analysis from images based on a backpropagation artificial neural network. *Nondestructive Testing and Evaluation* 23(4), 273–283 (2008)
4. de Albuquerque, V.H.C., da Silva, C.C., de Menezes, T.I.S., Farias, J.P., Tavares, J.M.R.S.: Automatic evaluation of nickel alloy secondary phases from sem images. *Microscopy Research and Technique* 74(1), 36–46 (2011)
5. Cieslak, M., Headley, T., Romig, A.: The welding metallurgy of hastelloy alloys c-4, c-22, and c-276. *Metallurgical and Materials Transactions A* 17, 2035–2047 (1986)
6. Collobert, R., Bengio, S.: Svmkernel: support vector machines for large-scale regression problems. *The Journal of Machine Learning Research* 1, 143–160 (2001)
7. Cormen, T., Leiserson, C., Rivest, R.: *Introduction to Algorithms*. MIT, Cambridge (1990)
8. Falcão, A.X., Stolfi, J., Lotufo, R.A.: The image foresting transform theory, algorithms, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26(1), 19–29 (2004)
9. Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.: LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research* 9, 1871–1874 (2008)
10. Papa, J.P., Falcão, A.X.: A new variant of the optimum-path forest classifier. In: Bebis, G., Boyle, R., Parvin, B., Koracin, D., Remagnino, P., Porikli, F., Peters, J., Klosowski, J., Arns, L., Chun, Y.K., Rhyne, T.-M., Monroe, L. (eds.) *ISVC 2008, Part I. LNCS*, vol. 5358, pp. 935–944. Springer, Heidelberg (2008)
11. Papa, J.P., Falcão, A.X., Suzuki, C.T.N.: Supervised pattern classification based on optimum-path forest. *International Journal of Imaging Systems and Technology* 19(2), 120–131 (2009)
12. Papa, J.P., Suzuki, C.T.N., Falcão, A.X.: *LibOPF: A library for the design of optimum-path forest classifiers* (2009), software version 2.0 available at <http://www.ic.unicamp.br/~afalcao/LibOPF>

13. Papa, J.P., Falcão, A.X.: A learning algorithm for the optimum-path forest classifier. In: Torsello, A., Escolano, F., Brun, L. (eds.) GbRPR 2009. LNCS, vol. 5534, pp. 195–204. Springer, Heidelberg (2009)
14. Perricone, M.J., Dupont, J.N.: Effect of composition on the solidification behavior of several ni-cr-mo and fe-ni-cr-mo alloys. *Metallurgical and Materials Transactions A* 37(4), 1267–1280 (2006)
15. Rocha, L.M., Cappabianco, F.A.M., Falcão, A.X.: Data clustering as an optimum-path forest problem with applications in image analysis. *International Journal of Imaging Systems and Technology* 19(2), 50–68 (2009)

Nonlinear Dynamical Analysis of Magnetic Resonance Spectroscopy Data

Alejandro Chinaa

Departamento de Física Fundamental, Facultad de Ciencias UNED,
Paseo Senda del Rey *n*°9, 28040 Madrid, Spain

Abstract. Nuclear Magnetic Resonance (NMR) Spectroscopy is a rapidly developing technique that measures chemicals within the brain without removing tissue or blood samples. Furthermore, it is an important tool for performing non-invasive quantitative assessments of brain tumour glucose metabolism. The principles underlying this technique have been successfully used to produce high quality images of neuroanatomy and disease processes. Unfortunately, current diagnosis techniques ignore the dynamic aspects of these signals. It is largely believed that temporal variations of NMR Spectra are simply due to noise or do not carry enough information to be exploited by any reliable diagnosis procedure. In this paper, we investigate the underlying characteristics of these signals using some complexity measures in combination with information theoretic concepts. The dynamics of these signals are further analyzed using elements from the theory of nonlinear dynamical systems. Furthermore, we show that they exhibit rich chaotic dynamics suggesting the encoding of metabolic pathway information.

Keywords: NMR Spectroscopy, medical diagnosis, nonlinear dynamics, information theory.

1 Introduction

The last decade has seen a rise in the application of proton NMR spectroscopy techniques, fundamentally in fields such as biological research [21], [23] and clinical diagnosis [22], [24]. The main goal within the biological research field is to achieve a deep understanding of metabolic processes that may lead to advances in many areas including clinical diagnosis, functional genomics, therapeutics and toxicology. From a clinical diagnosis point of view, a large number of proton NMR spectroscopy applications have targeted the human brain. This is mainly due to the fact that proton NMR spectroscopy is a non-invasive technique, which is particularly important in this part of the body where clinical surgery or biopsy is more delicate than in other areas. In this paper, the focus is on proton NMR brain spectroscopy. The process of clinical diagnosis involves the analysis of spectroscopy signals obtained from a well-defined cubic volume of interest (single voxel experiment) in a specific region of the brain during a pre-defined time frame (acquisition time). Two acquisition methods are commonly used, namely

point resolved spectroscopy (PRESS) [5] or stimulated echo acquisition mode (STEAM) [20]. The clinical researcher has a strong interest in understanding the role of metabolites under normal and pathological conditions, which is not possible without a deep knowledge of the interactions between them. These interactions are based on a wide set of chemical reactions which are organized into metabolic pathways. Specifically, the chemical reactions involve the transformation of one metabolite into another through a series of steps catalysed by a sequence of enzymes. Surprisingly, the most relevant characteristic of metabolic pathways [9], [15] is their universality, i.e. the fact that metabolic pathways are very similar even across quite different species. It is noteworthy that the concept of metabolic pathways is inherently associated with a dynamic context [4], [8]. Therefore, extracting information about metabolic pathways from complex biological data sets like those generated through NMR spectroscopy is a challenging research task that requires consideration of the dynamic aspects of these signals.

However, current diagnosis techniques ignore the dynamic aspects of these signals. It is largely believed that the information content of temporal variations of NMR Spectra is minimal. Thus, current diagnosis procedures are constrained to empirical observations extracted from a single averaged spectrum.

The rest of this paper is organized as follows: In the next section, the problems and difficulties associated with the processing of NMR signals are presented. In section 3 a formal characterization of NMR-based data using complexity measures and information theoretic concepts is introduced. Section 4 focuses on the dynamic aspects of NMR spectral signals. Finally, section 5 provides a summary of the present study and some concluding remarks.

2 Problems Associated with NMR Data

Generally speaking, the spectral signals associated with brain metabolites are characterized by one or more peaks at certain resonance frequencies. Furthermore, the molecular structure of a particular metabolite is reflected by a typical peak pattern. In addition, it is also important to note that the area (amplitude) of a peak is proportional to the number of nuclei that contribute to it and therefore to the concentration of the metabolite to which the nuclei belong. Most of the metabolites have multiple resonances many of which are split into multiplets as a result of homonuclear proton scalar coupling. This fact is particularly true of proton NMR spectroscopy at clinical field strengths (from 1.5 up to 3 Teslas), where the whole spectrum occupies a narrow frequency range, normally from -0.8 up to 4.3 parts per million (ppm hereafter) in brain NMR spectroscopy, resulting in significant overlap of peaks from different metabolites. In addition, the response of coupled spins is strongly affected by the acquisition parameters of the NMR sequence (e.g. radio frequency pulses employed and the time intervals set between them [3]). Furthermore, additional difficulties are caused by the presence of uncharacterized resonances from macromolecules or lipids. Specifically, in a typical NMR profile a large number of the resonances may be unassigned, particularly for low level or partially resolved signals. This is further

complicated by small but significant sample to sample variations in the chemical shift position of signals, produced by effects such as differences in pH and ionic strength. This kind of positional noise remains a problem as information coming from a given metabolite contaminates the spectral dimensions containing information from other metabolites. In other words, due to this phenomenon the resonance frequency of certain metabolites can suffer slight variations from sample to sample.

However, the common factor to all the abovementioned techniques is that they completely ignore the inherent dynamics of NMR spectroscopy signals. Unfortunately, it is largely believed that temporal variations of NMR Spectra are simply noise or do not carry enough information to be exploited by any reliable diagnosis procedure.

3 Complexity Measures and Information Theory

In the following sections, we are mainly concerned with the analysis of time series of spectra. Specifically, as opposed to standard single-voxel spectroscopy experiments where the resulting spectral signals are averaged in what follows we consider the entire set of frames obtained after the NMR acquisition process. Accordingly, for experimental purposes we used data collected from 11 healthy patients of ages ranging from 25 up to 45, with a mean of 31.45 years. The data was collected from different brain regions and with approximately equal voxel sizes. The acquisition time was approximately equal to 5 minutes for all patients, using a total echo time (TE) equal to 23ms and a repetition time (TR) of 1070ms. Furthermore, the data was pre-processed using the whitening transformation [1].

For a given nucleus, the amplitude and frequency are the most specific parameters of individual resonances in NMR spectra. In the case of scalar coupled spins, the resonances are split into several smaller resonances according to a well-defined pattern which is dependent on the other coupled spins. A detailed compilation of proton chemical shifts of the groups (i.e. methyl, methylene, amines etc) which compose brain metabolites is presented in table 1. This table was built by decomposing the chemical shift range associated with the 35 brain metabolites presented in [7] into bins of 0.1 ppm resolution width. From inspection of the table, it can be observed that there are regions which are more populated with molecular group contributions than others. In particular, the interval [3.0, 4.0] supports the biggest amount of group contributions when compared to the rest of the intervals. Furthermore, we can appreciate certain intervals where apparently there is no contribution at all of any of the molecular groups. For instance, the interval [-0.8, 0.9] constitutes a clear example of this. Taking into account these observations, it is reasonable to think that increased complexity should be observed in metabolic signals associated with the intervals where there are more contributions of molecular groups. In addition, it would be desirable to quantify the complexity of any such increment. A natural way of assessing the complexity associated with metabolic signals is to use a complexity measure such as the fractal dimension [2], [17].

Table 1. Contributions of molecular groups associated with the 35 brain metabolites within the chemical shift interval [-0.8, 4.4]

[-0.9,-0.8]	[-0.8,-0.7]	[-0.7,-0.6]	[-0.6,-0.5]	[-0.5,-0.4]	[-0.4,0.3]	[-0.3,-0.2]	[-0.2,-0.1]	[-0.1,0.0]	
[0.0,0.1]	[0.1,0.2]	[0.2,0.3]	[0.3,0.4]	[0.4,0.5]	[0.5,0.6]	[0.6,0.7]	[0.7,0.8]	[0.8,0.9]	[0.9,1.0]
[1.0,1.1]	[1.1,1.2]	[1.2,1.3]	[1.3,1.4]	[1.4,1.5]	[1.5,1.6]	[1.6,1.7]	[1.7,1.8]	[1.8,1.9]	CH_3
CH_3			CH_3 CH_3	CH_3				CH_2 CH_2	CH_3
[2.0,2.1]	[2.1,2.2]	[2.2,2.3]	[2.3,2.4]	[2.4,2.5]	[2.5,2.6]	[2.6,2.7]	[2.7,2.8]	[2.8,2.9]	[2.9,3.0]
CH_3 CH_3	CH_2 CH_2 CH_2 CH_2 CH_2	CH_2 CH	CH_2 CH_2 CH_2 CH_2 CH_3 CH_2	CH_2 CH_2 CH_2	CH_2 CH_2	CH_2 CH_2		CH_2	CH_2 CH_2 CH_2 CH_2 CH_2
[3.0,3.1]	[3.1,3.2]	[3.2,3.3]	[3.3,3.4]	[3.4,3.5]	[3.5,3.6]	[3.6,3.7]	[3.7,3.8]	[3.8,3.9]	[3.9,4.0]
CH_2 $N(CH_3)$ CH_2 CH_2 $N(CH_3)$ CH_2	$N(CH_3)_3$ CH_2 CH_2 CH_2 CH_2 CH_2	$N(CH_3)_3$ CH CH_2 CH CH_2 $N(CH_3)$ CH_2 CH_2 CH_2	CH	CH_2 CH_2	CH_2 CH_2 CH_2 CH_2 CH CH CH CH	CH_2 CH_2 CH_2 CH_2 CH_2 CH CH CH_2	CH CH CH CH_2 CH CH	CH CH_2 CH_2 CH	CH_2 CH CH_2 CH CH CH_2 CH_2 CH_2 CH
[4.0,4.1]	[4.1,4.2]	[4.2,4.3]	[4.3,4.4]						
CH_2 CH CH		CH_2 CH	CH CH_2						

The fractal dimension is a parameter which measures the intrinsic dimension or degrees of freedom of a data set. Furthermore, it can be defined as the dimension of the sub-manifold structure of the data. More specifically, it can be considered as a similarity dimension, in other words, how a data set object remains statistically similar independently of the scale of observation. To this end, we conducted a series of experiments. Firstly, we computed the fractal dimension of the data set object associated with the interval [0.3, 0.4]. This meant working in a space of about 20 dimensions (0.1 ppm resolution width). Figure 1 consists of the graph obtained when applying the algorithm presented in [2] to the data set object associated with the interval [0.3, 0.4]. The vertical axis represents the logarithm of the number of hypercubes of size equal to r . By inspection of the graph, we can observe two regions which can be differentiated by the slope of the graph. The first region is the smallest one and goes from $\log(1/r) = 0$ (hypercube size equal to 1) up to $\log(1/r) \approx 3$. For this interval the slope is approximately 1.6. Finally, the last interval has a slope of approximately 1. Therefore, the fractal dimension is approximately 1.6. It is important to note that we have adopted a conservative approach, instead of averaging the slopes of the graph we take the worst case behaviour of the data set object measured in terms of complexity. Therefore, the samples would be in a volume of about 2 dimensions. That is, two degrees of freedom are responsible for the observed complexity.

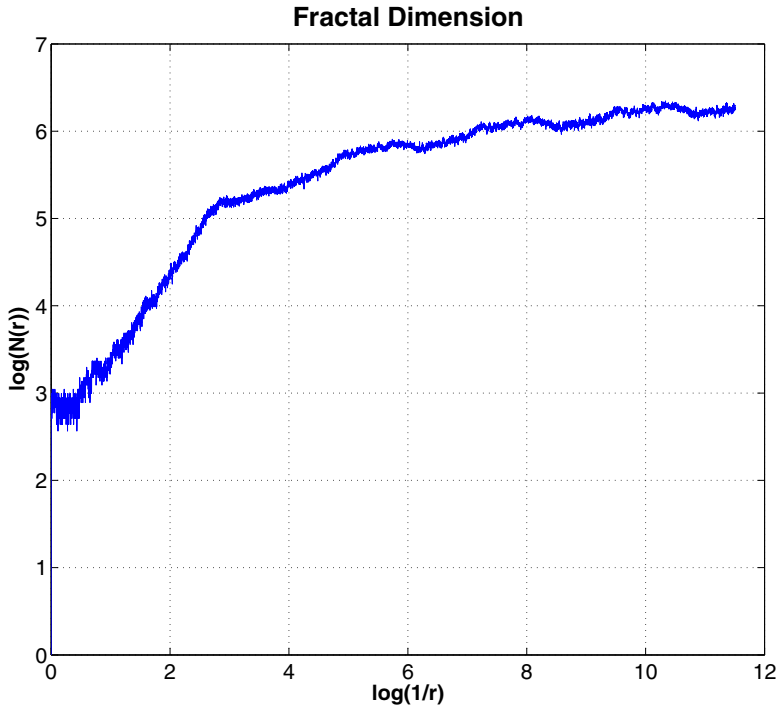


Fig. 1. Fractal dimension computation for the interval [0.3, 0.4] ppm using bins of 0.1 ppm width and computed for the experimental data set

If we proceed in a similar way but now taking as a reference for the computation the interval [3.9, 4.0] we obtain fractal dimension is approximately 4.5, which means that the samples would fit in a volume of about 5 dimensions. Furthermore, the fact that the region has the highest amount of molecular group contributions does not lead to an explosion in the complexity associated with this region but rather a slight increase in complexity - there are five degrees of freedom. At this point, it is important to emphasize the fact that the fractal dimension of a random process is infinite. Therefore, we would observe regions of the curve with an infinite slope. Additionally, we performed the same computation for each of the bins associated with the interval [-0.8, 4.3], and we found that complexity is bounded within the integer interval [2,5]. In turn, we performed a PCA [10] analysis of each of the bins composing the range of interest of [-0.8, 4.3] ppm associated with brain metabolites using bins of 0.1 ppm resolution width. Furthermore, we performed the PCA transformation keeping 98% inertia. In other words, allowing only 2% information loss.

Figure 2 depicts the results of the transformation. The horizontal axis represents the chemical shift scale in parts per million (ppm) while the vertical axis represents the dimensionality reduction achieved (number of dimensions needed for the above PCA parameterization). From inspection of the graph we can account for three different behaviours.

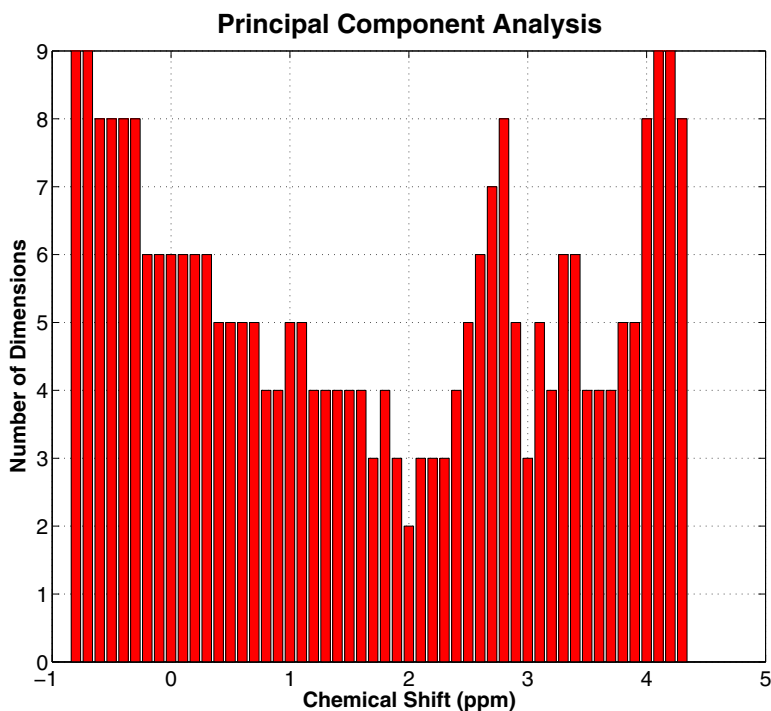


Fig. 2. Principal component analysis for the interval $[-0.8, 4.3]$ ppm using bins of 0.1 ppm width and computed from data set

Firstly, regions where contributions of molecular groups are either minimal (i.e., one or two group contributions) or inexistent and the PCA needs a considerable number of dimensions (8 to 9 dimensions) to explain the variance of the data (i.e. the interval $[-0.8, 0.3]$ and the interval $[4.0, 4.4]$). This behaviour might suggest the existence of noise in these bands. Indeed, the PCA transform cannot reduce the dimensionality of a random process. However, this hypothesis is in contradiction with the analysis of complexity obtained before that indicated the presence of an underlying dynamical system with few degrees of freedom. The PCA transform is limited by virtue of being a linear technique. Therefore, a plausible hypothesis for explaining the results is to consider the existence of non-linear correlations between the metabolic signals associated with these regions.

Secondly, regions where the contributions of molecular groups are minimal or inexistent but the PCA achieves a considerable reduction in dimensionality (i.e. the interval $[0.8, 1.8]$). For this region we can observe a certain discrepancy between the degrees of freedom obtained with the PCA and the with the fractal dimension computation. This fact seems to justify again the hypothesis for the existence of non-linear correlations since the PCA seems to be overestimating the true dimensionality of the data sets. Finally, regions where there is a large amount of molecular group contributions (around 7 contributions on average)

and the PCA achieves a considerable dimensionality reduction (4 to 5 dimensions). For instance, the intervals $[3.5, 4.0]$ and $[2.9, 3.2]$. Again, the hypothesis of non-linear correlations would also fit the description of results presented for the third observation.

In order to shed some light on the previous results, we computed the entropy [19] associated with each of the metabolic signals throughout the range $[-0.8, 4.3]$. The idea was to consider each metabolic signal as a realization of an unknown stochastic process. It is important to remember that each metabolic signal here is a time series composed of spectral amplitudes.

Figure 3 illustrates the results of the entropy computation. The horizontal axis represents the chemical shift scale in parts per million (ppm) while the vertical axis represents entropy in nats. From inspection of the graph, it is easy to deduce that the values of entropy present strong oscillations, even for metabolic signals whose associated resonance frequencies are contiguous or very close in the spectrum. These oscillations account for the complexity of the information distribution. Moreover, by analyzing the values of entropy reached, we can sub-divide the graph into three logical regions. Firstly, the region from $[2.0, 4.0]$ ppm is the region where average entropy is higher compared to the rest of the intervals under consideration. Furthermore, this region is characterized by the fact that it has the

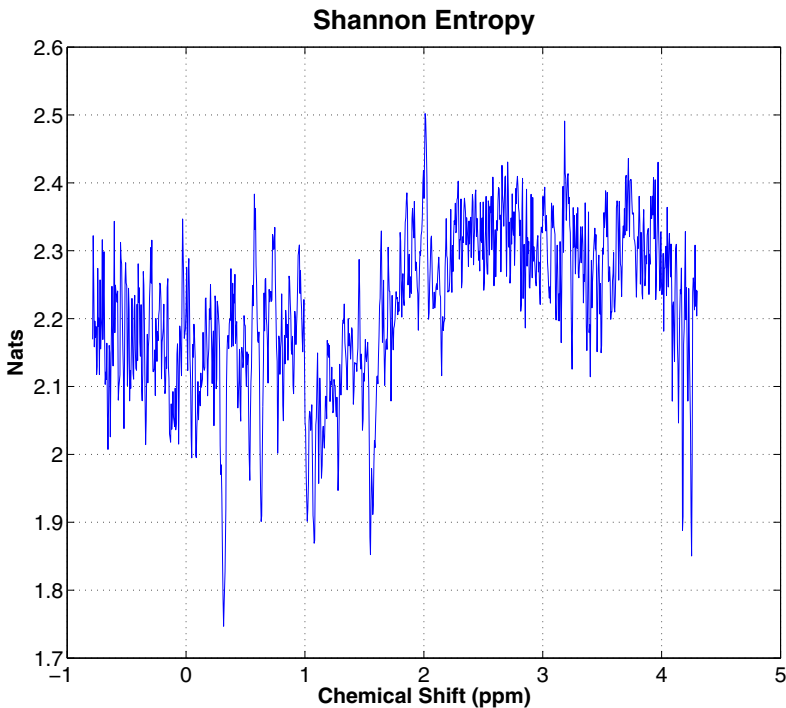


Fig. 3. Entropy of the set of metabolic signals (stochastic processes) associated with the interval $[-0.8, 4.3]$ ppm computed for data set

highest amount of group contributions (see table 1 for details). Therefore, as may be expected, the degree of randomness (i.e. disorder) or uncertainty associated with such stochastic processes is on average higher than that of other regions of the spectra. However, it is important to note that having higher entropy values also means that on average whenever we observe such processes, the amount of information conveyed by each realization of the processes is also higher.

Secondly, the region from $[-0.8, 1.5]$ is the region where average entropy reaches lower values than in other regions of the spectrum. Indeed, global minimum entropy is reached within this interval. In addition, as opposed to region (1), this region is characterized by the absence of molecular group contributions. Following similar reasoning to that sketched above, whenever we observe any of the stochastic processes associated with this region, the average information conveyed per observation is lower as compared to those of region (1). Therefore, these processes (i.e. time series) are more predictable. Finally, the regions $[1.5, 2.0]$ and $[4.0, 4.3]$ constitute a special case as they correspond to regions also characterized by the complete absence of, or the presence of very few molecular group contributions but the values of entropy reached in these regions are comparable on average to those of region (1).

Concerning interval $[1.5, 2.0]$, it is important to note that the values of entropy show an increasing tendency throughout this interval. Indeed, the global maximum entropy is reached in sub-interval $[2.0, 2.1]$ corresponding to the N-Acetyl aspartate (NAA) resonance. This is particularly interesting as we have a region where, firstly the degree of randomness is the highest compared to the rest of the regions associated stochastic processes which are information-rich, but at the same time we have an underlying system which has few degrees of freedom (only three).

4 Nonlinear Dynamical Analysis

The analysis performed in section 3 shown that the highest entropy values correspond in general to regions where there is a significant amount of molecular group contributions. Conversely, regions with little or no molecular group contributions are characterized by lower entropy values. However, this empirical rule is only partially true as it shows certain exceptions. Specifically, the unusual relationship observed between information distribution, degree of randomness and the underlying complexity shown by the data. This puzzle is solved, as it is demonstrated shortly, if we think in terms of a chaotic system. A chaotic system is a deterministic system in the sense that its operation is governed by fixed rules, yet such a system with only a few degrees of freedom can exhibit behaviour so complex that it looks random.

In order to visualize the nonlinear properties of the temporal NMR data, we used Poincaré plots [9, 18] to draw the phase space of metabolic signals. In particular, in order to simplify the problem we worked for representational purposes with uni-dimensional stochastic processes. Specifically, from the matrix associated with our experimental data set, we drew the phase space of the metabolic signal (dimension) associated with a given metabolite having the highest entropy value.

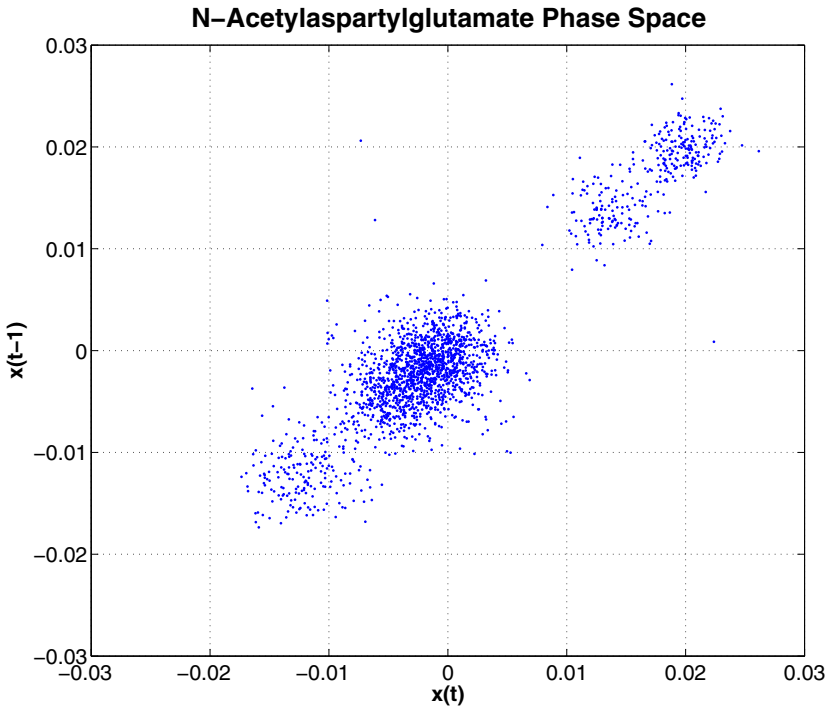


Fig. 4. Phase Space representation associated with the NAAG time series

Figures 4 and 5 represent the phase space of the time series associated with the excitatory neurotransmitters N-Acetylaspartylglutamate and glutamate, respectively, for a unitary delay. N-Acetylaspartylglutamate (NAAG hereafter) is a dipeptide of N-substituted aspartate and glutamate. It has been suggested that it is involved in excitatory neurotransmission as well as being a source of glutamate, although its function remains to be clearly established. For representation purposes we chose the singlet resonance at 2.04 ppm of the *acetyl* - CH_3 protons. In addition, glutamate is an amino acid with an acidic side chain. It is the most abundant amino acid found in the human brain. It is known to act as an excitatory neurotransmitter, although it is believed to have other functions too. It has two methylene groups and a methine group which are strongly coupled. In this case, we again opted for phase space representation, the resonance having maximum entropy value which occurs at 3.7433 ppm. The structure of the graphs seems to confirm the hypothesis that we are in the presence of a chaotic system. The most important point to note is that the processes represented are not random. In fact, they present the typical aspect of the phase space of a chaotic attractor. Random processes are characterized by a randomly distributed phase space (e.g. points uniformly distributed around the phase space) with uncorrelated data points. However, we observe here a specific structure that reflects the ergodicity of the underlying stochastic process. It is important to remember that the standard deviation of the data points perpendicular to the line

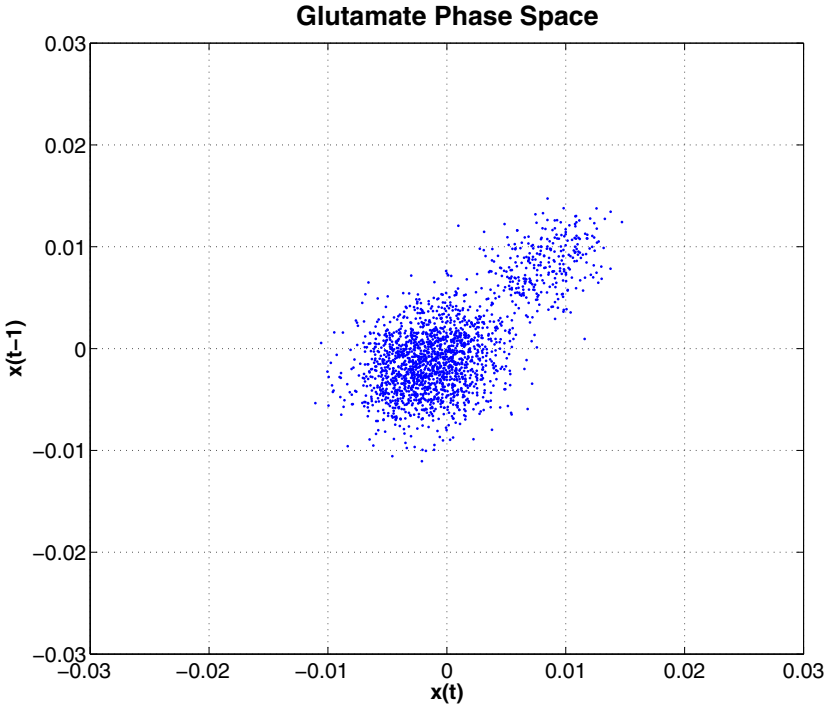


Fig. 5. Phase Space representation associated with glutamate time series

of identity describes short-term variability of the stochastic process. Conversely, the standard deviation along the line of identity describes long term variability. Therefore, it is easy to deduce that the short-term and long-term variability of NAAG process are higher when compared to those of glutamate. Furthermore, its phase diagram suggests the existence of four attracting regions of phase space trajectories as compared to the two regions that can be observed in the glutamate phase space, showing that the former is apparently more complex or at least information-rich.

On the other hand, it is important to note that we are dealing with finite amounts of data. Therefore, in order to better exploit the available information, using a unitary delay for computing the phase space diagrams can hide important information about the structure of the chaotic process under consideration. Unitary delays lead to delay vectors which are all concentrated around the diagonal (because of the correlations) in the embedding space and thus the structure perpendicular to the diagonal may not be visible. Let us denote by $y(t)$ the time series associated with any of the metabolites resonances. The optimal choice for the embedding delay τ is that value of t that makes $y(t)$ and $y(t - \tau)$ independent, i.e. having no correlation with each other.

As stated in [6] this requirement is best satisfied by using the particular t for which the mutual information between $y(t)$ and $y(t - \tau)$ attains its first minimum. Taking these considerations into account we computed the optimal embedding

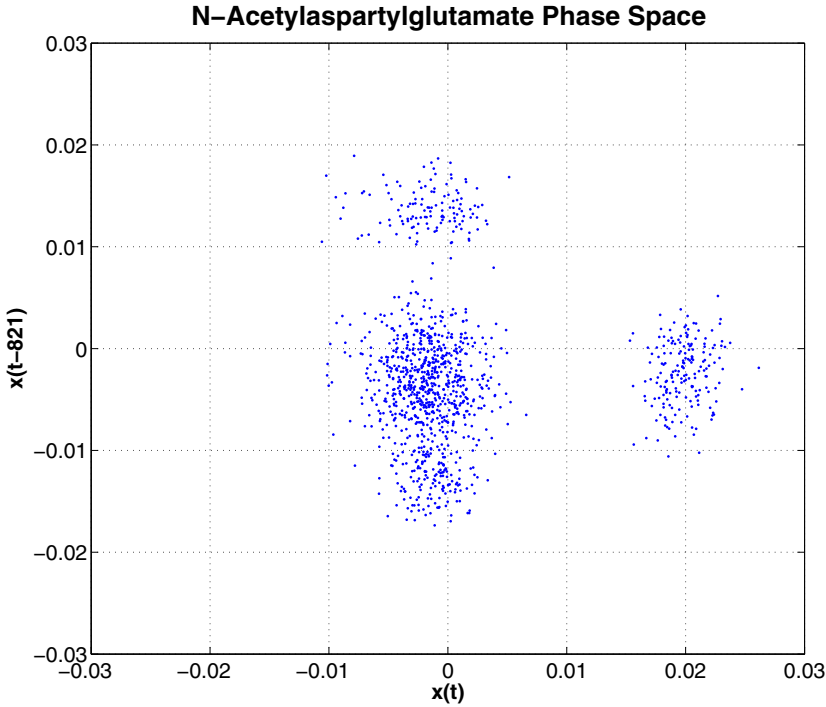


Fig. 6. Phase Space representation of the NAAG time series using the optimal embedding delay computed with the mutual information procedure [6]

delay for the time series associated with NAAG obtaining an embedding delay value of 821.

Figure 6 displays the phase diagram of NAAG but using the delay coordinates computed with the method described above based on mutual information. As can be observed, the shape of the phase space changes substantially when using the appropriate delay coordinates. In particular, we observe three completely differentiated attracting regions. Moreover, in order to give formal proof of the chaotic behaviour we computed the maximum Lyapunov exponent of the NAAG time series using the method proposed in [12], and obtaining a value of $\lambda_{max} \approx 0.014$, thereby confirming the hypothesis of chaos. Although not shown here, similar proofs can be provided using the time series associated with the other metabolites. It is important to remember that a chaotic process is defined as a process generated by a nonlinear deterministic system with at least one positive Lyapunov exponent.

To summarize, we have proved that temporal variations of NMR metabolic signals are information-rich, by revealing their chaotic structure. Furthermore, we have also seen that the amplitude of the spectrum signal associated with a given metabolite is proportional to the concentration of that metabolite. Trajectories induced by the attractors depicted in figures 4 and 5, in fact reflect concentration changes. Furthermore, such concentration changes are not random but they

follow a pattern which is specific to the metabolite under consideration. In addition, the dataset used throughout the previous analysis corresponds to NMR data coming from different individuals using an acquisition time big enough to observe inhibitory and excitatory neurotransmission processes. Taking these considerations into account together with the fact that the set of chemical reactions that happen in living organisms are universal, in particular metabolic pathways, it is a plausible hypothesis to state that the observed chaotic behavior inherent to metabolic signals is in fact encoding metabolic pathway information. At this point, it is important to remember that chaos is present in many human physiological processes [13] such as the cardio-respiratory system, the perception and motor control system, and voice, amongst others.

However, in order to extract such information we have to exploit the full multidimensional information associated with each metabolite and to use nonlinear filter techniques [14], [16]. Nonlinear noise reduction consists of phase space reconstruction techniques that do not rely on frequency information in order to define the distinction between signal and noise. Instead, structure in the reconstructed phase space will be exploited. It is important to remember that chaotic trajectories display wide-band spectra where power decreases with the inverse of frequency. These spectra differentiate chaos from noise: white noise displays a uniform distribution of frequencies across the spectrum. Nonlinear noise reduction takes into account that nonlinear signals will form curved structures in delay space. In particular, noisy deterministic signals form blurred-out lower dimensional manifolds. Nonlinear phase space filtering tries to identify such structures and project onto them in order to reduce noise. Although further research must be carried out, these preliminary results have shown the possibility of mapping empirical NMR spectroscopy data to metabolic pathways. This fact would contribute not only to the development of early tumour detection techniques but to clarifying the high degree of information that is missing from our current understanding of complex biological systems.

5 Conclusions

In this paper we have investigated the application of machine learning techniques and chaos theory to characterize magnetic resonance spectroscopy data. Throughout this paper we have focused explicitly on the characterization of dynamic brain NMR data. We have presented a formal description of the problems associated with NMR-based data in terms of its application context in the field of clinical diagnosis research. Specifically, we reviewed the most common problems derived from proton scalar coupling effects, as well as those derived from differences in pH and ionic strength. Furthermore, we put especial emphasis on the fact that despite considerable progress in clinical diagnosis methodologies, current diagnosis techniques make the assumption that temporal correlations from sample to sample between metabolic signals do not carry information at all. Moreover, in order to understand the underlying structure of dynamic NMR-based data we have performed an analysis based on information theoretic concepts and

nonlinear dynamical systems theory. The results revealed an information-rich structure as shown by the chaotic nature of dynamic NMR data. Furthermore, we argued that they actually encode metabolic pathway information. The main contribution of the present study is to invalidate the traditional view that disregards the dynamic aspects of NMR data as being devoid of information. We can conclude, firstly that a deep understanding of the complex relationships between metabolites under normal and pathological conditions cannot be conceived without taking into account their dynamical interactions. Furthermore, the framework presented here opens the possibility for efficiently modelling the dynamics of deep molecular pathways with high levels of noise and relatively small experimental data sizes.

Acknowledgments. The author would like to thank Dr. Elka Korutcheva for her support in the dissemination of this research and also for her constructive and helpful comments.

References

1. Bishop, C.M.: *Neural Networks for Pattern Recognition*. Oxford University Press, Oxford (1995)
2. Blayo, F., Cheneval, Y., Guerin-Dugue, A., et al.: Enhanced Learning for Evolutionary Neural Architecture, ESPRIT Basic Research Project Number 6891, Deliverable R3-B4-P, Task B4 (Benchmarks), pp. 11–22 (1995)
3. Cloarec, O., Dumas, M.E., Craig, A., et al.: Statistical Total Correlation Spectroscopy: An Exploratory Approach for Latent Biomarker Identification from Metabolic ^1H NMR Data Sets. *Anal. Chem.* 77, 1282–1289 (2005)
4. De Jong, H.: Modeling and Simulation of Genetic Regulatory Systems: A Literature Review. *J. Computat. Biol.* 9, 67–103 (2002)
5. Frahm, J., Hanioké, W., Merboldt, K.D.: Transverse coherence in Rapid FLASH NMR Imaging. *J. Magn. Reson.* 72, 307–314 (1987)
6. Fraser, A.M., Swinney, H.L.: Independent coordinates for strange attractors from mutual information. *Phys. Rev. A* 33, 1134–1140 (1986)
7. Govindaraju, V., Young, K., Maudsley, A.A.: Proton NMR Chemical Shifts and Coupling Constants for Brain Metabolites. *NMR Biomed.* 13, 129–153 (2000)
8. Hasty, J., McMillen, D., Isaacs, F., et al.: Computational Studies of Gene Regulatory Networks: in numero molecular biology. *Nat. Rev. Gene.* 2, 268–279 (2001)
9. Hegger, R., Kantz, H., Schreiber, T.: Practical Implementation of Nonlinear Time Series Methods: The TISEAN package. *CHAOS* 9, 413–436 (1999)
10. Jolliffe, I.T.: *Principal Component Analysis*. Springer, New York (1986)
11. Kanehisa, M., Goto, S.: KEGG: Kyoto Encyclopedia of Genes and Genomes. *Nucleic Acids Res.* 28, 27–30 (2000)
12. Kantz, H.: Quantifying the Closeness of Fractal Measures. *Phys. Rev. E* 49, 5091–5097 (1994)
13. Kantz, H., Kurths, J.: *Nonlinear Analysis of Physiological Data*. Springer, New York (1998)
14. Kantz, H., Schreiber, T., Hoffmann, I., et al.: Nonlinear noise reduction: a case study on experimental data. *Phys. Rev. E* 48, 1529–1538 (1993)

15. Karp, P.D., Riley, M., Paley, S.M., et al.: The MetaCyc Database. *Nucleic Acids Res.* 37, 59–61 (2002)
16. Kostelich, E.J., Schreiber, T.: Noise reduction in chaotic time series data: A survey of common methods. *Phys. Rev. E* 48, 1752–1763 (1993)
17. Le Méhauté, A.: *Les Géométries Fractales*, Hermès, Paris (1990)
18. Lin, C., Wang, J., Chung, P.: Mining Physiological Conditions from Heart Rate Variability Analysis. *IEEE Computational Intelligence Mag.* 5(1), 50–58 (2010)
19. McKay, D.J.C.: *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, Cambridge (2003)
20. Nelson, S.J., Brown, T.R.: A Method for Automatic Quantification of One-dimensional Spectra with low Signal-to-noise Ratio. *J. Magn. Reson.* 75, 229–243 (1987)
21. Nicholson, J.K., Lindon, J.C., Holmes, E.: Understanding the metabolic responses of living systems to pathophysiological stimuli via multivariate statistical analysis of biological NMR spectroscopic data. *Xenobiotica* 29(11), 1181–1189 (1999)
22. Passe, T.J., Charles, H.C., Rajagopalan, P., Krishnan, K.R.: Nuclear Magnetic Resonance Spectroscopy: A review of Neuropsychiatric Applications. *Prog. Neuro-Psychopharmacol. And Biol. Psychiat.* 19, 541–563 (1995)
23. Raamsdonk, L.M., Teusink, B., Broadhurst, D., et al.: A functional genomics strategy that uses metabolome data to reveal the phenotype of silent mutations. *Nat. Biotechnol.* 19(1), 45–50 (2001)
24. Szabo De Edelenyi, F., Rubin, C., Estevez, F., et al.: A new Approach for Analyzing Proton Magnetic Resonance Spectroscopic Images of Brain Tumors: Nosologic Images. *Nat. Med.* 6, 1287–1289 (2000)

A Shared Parameter Model for Gesture and Sub-gesture Analysis

Manavender R. Malgireddy, Ifeoma Nwogu, Subarna Ghosh,
and Venu Govindaraju

Computer Science and Engineering
SUNY at Buffalo, NY, USA
{mrm42,inwogu,sghosh7,govind}@buffalo.edu

Abstract. Gesture sequences typically have a common set of distinct internal sub-structures which can be shared across the gestures. In this paper, we propose a method using a generative model to learn these common actions which we refer to as sub-gestures, and in-turn perform recognition. Our proposed model learns sub-gestures by sharing parameters between gesture models. We evaluated our method on the Palm Graffiti digits-gesture dataset and showed that the model with shared parameters outperformed the same model without the shared parameters. Also, we labeled different observation sequences thereby intuitively showing how sub-gestures are related to complete gestures.

1 Introduction

Recent advances in computer vision and machine learning have led to a new modeling paradigm where high-level problems can be modeled using combinations of lower-level segmental units. Such units can be learned from large datasets and represent the universal set of alphabets to fully describe a vocabulary. For example, in a high-level problem such as speech recognition, a phoneme is defined as smallest segmental unit employed to form an utterance (speech vector). Similarly in handwriting recognition, a word can be represented as a sequence of strokes, where each stroke is the smallest segmental unit. Also, in object labeling in images, the bag-of-words (or bag-of-features) technique learns the set of small units required to segment and label the object parts in an image.

Motivated by the successes of this modeling technique in solving general high-level problems in computer vision, we define a gesture as a sequence of contiguous sub-gestures, where the sub-gesture is a discrete unit that can be identified in a gesture stream. Sub-gestures are separate and individual, and occur in temporal order. For example, in a natural setting, when a person waves goodbye, the sub-gestures involved could be (i) raising a hand from rest position to a vertical upright position; (ii) moving the arm from right to left; and (iii) moving the arm from left to right. The entire gesture therefore consists of the first sub-gesture occurring once and the second and third sub-gestures occurring multiple times.

In this paper, we present a generative shared parameter model for learning the underlying latent structure that is common to a series of gestures. We quantitatively demonstrate how the use of shared parameters outperforms a similar, generative but non-shared model, and also show how the learned latent structure of the sub-gesture correspond to discrete, consistent actions performed by the user. It is important to note that the goal of our study is not primarily to perform gesture classification, but rather to develop a framework for identifying the relevant sub-gestures, given a large gesture data corpus. This type of analysis can be very useful in analyzing gestures in a natural (or other unfamiliar) setting where the true number of gestures is not necessarily known beforehand. Hence, we present our framework using a generative shared parameters model.

The main contribution of this paper is the presentation of a generative technique to learn about the implicit nature of sub-gestures, which will in turn help in understanding gestures more intuitively. This becomes more important as the field begins to perform more analysis on gestures in natural rather than experimental settings.

1.1 Related Work

Gesture modeling, spotting and classification have been studied extensively and there is a wide range of related work in the literature. Some related reviews and surveys include [4][11][15] of which the most recent was published in 1999.

More recently, the use of discriminative classifiers such as Hidden Conditional Random Field model(HCRF), Latent-Dynamic Conditional Random Field (LD-CRF) [10] have proven quite successful for modeling gestures, when classification is the primary task at hand. Although in this paper, we focus our attention more on better understanding the relationship between gestures and their underlying representative sub-gestures, as a first step towards large-scale gesture understanding in natural settings.

Similar generative models used in gesture analysis include a technique to recognize head nods and head shakes using two Hidden Markov Models (HMMs) whose features are 2D coordinate points obtained results from an eye gaze tracker [6]. Sign language gesture analysis [2][13] as well as arm gesture recognition [3] also use HMMs and related models. Many of these earlier studies of gesture recognition were based on the one-versus-all models, where a separate model is trained for each gesture class e.g. HMMs and its variants [2][8][16]. Recent advances in gesture recognition research suggest that multiclass (one model for all gesture classes) models as in [14] jointly learn the best discriminative structure among gestures. As the authors in [14] showed, Hidden Conditional Random Fields (HCRF) provide a good framework for modeling a gestures as a combination of sub-gestures by sharing hidden states among the multiple gesture classes. However, this sharing is implicit and there is no way to explicitly define a sub-gesture sequence for a given gesture which might be useful for continuous gesture recognition. Moreover, HCRF training algorithms are computationally very expensive when compared to HMMs. We therefore propose a novel variant of HMM which combines the advantages of HCRF with the generative modeling

power of HMMS. The proposed approach models each gestures using one HMM, but shares the state parameters across all the HMMs, similar to the multiclass paradigm. This results in a computationally efficient framework for explicit gesture modeling and implicit sub-gesture modeling.

In many dynamic gesture recognition systems such as in [3], lower-level modules are employed to perform explicit spatial segmentation to extract the appearance features of the body parts to be tracked. Those features are then passed into the recognition module, which classifies the gesture (Pavlovic et al. provide a detailed review of such dynamic systems in [11]). These bottom-up methods rely heavily on the segmentation results, and in spite of the advances in person and body-parts detection (including hand and head detection), it is still a very challenging task in many real-life settings. Rather than requiring a perfect hand detection, in our approach, we assume that crude but fast existing hand detection methods can produce a relatively short list of candidate hand locations for every frame of the input sequence. We also assume that each short list always contains the true location of the gesturing hand. A similar assumption has been made by Sato et al. [12] as well as Alon et al. [1]. The differentiating factor between our employed method and those of [12] and [1] is the probabilistic selection of the most likely candidate. The main advantage of this paradigm is that it does away with the assumption that the gesturing hand has been well localized before testing is done, and it also achieves significant speed-ups by eliminating a large numbers of implausible hypotheses from consideration.

2 Feature Extraction

From each frame in a video, we extract multiple candidate hand locations using skin color and optical flow information. A face detector is run initially, followed by a skin detector which calculates the likelihood of each image pixel belong to skin. For the first few frames, if a face is not detected, we use the skin color model proposed by [5] to generate candidates. Once a face is detected [7], we calculate the mean and variance of the face pixels in normalized rg space and use them to calculate a skin likelihood image. A motion mask of the image is obtained by frame differencing. The skin likelihood image is multiplied with the motion mask to obtain hand likelihood image.

We obtain top K candidate hand locations by convolving a 40 rows x 30 columns mask with the hand likelihood image (similar to the mask size used in [1]). In this manner for every frame j we extract K candidate hand locations and for each hand location, we extract a 4D feature vector $O_{j,k} = (x_{j,k}, y_{j,k}, u_{j,k}, v_{j,k})$ comprising of location information (x, y) and the optical flow (u, v) averaged over the region K [9].

3 Model Description and Parameter Learning

We model gestures as a generative process. Rather than learning an HMM for each gesture, with each HMM having different state transition probability

distributions and observation densities, our proposed method learns a generative model such that the observation densities are shared among all the gestures and only the state transition probabilities are different for each gesture.

Let G be the number of gesture classes in the dataset and N be the number of hidden states in the model. An observation sequence is represented by $O = \{O_1, O_2, ..O_t,, O_T\}$, where O_t represents an observation at time t . Let $S = \{S_1, S_2,, S_N\}$ be the set of all states and we denote the state at time t as q_t . The state transition probability distribution for gesture g is $A^{(g)} = \{a_{ij}^{(g)}\}$ where

$$a_{ij}^{(g)} = P(q_{t+1} = S_j | q_t = S_i, g), \quad 1 \leq i, j \leq N$$

$$1 \leq g \leq G(1)$$

The observation probability distributions is given by $B = \{b_j\}$ where

$$b_j(O_t) = \mathcal{N}(o_t | \mu_j, \Sigma_j), \quad 1 \leq j \leq N. \tag{2}$$

Initial state probabilities for a gesture g is given by $\pi^{(g)} = \{\pi_i^{(g)}\}$ where

$$\pi_i^{(g)} = P(q_1 = S_i | g), \quad 1 \leq i \leq N. \tag{3}$$

The complete parameter set of the model is

$$\lambda = \{A^{(1)}, A^{(2)},, A^{(G)}, B, \pi^{(1)}, \pi^{(2)},, \pi^{(G)}, \} \tag{4}$$

3.1 Inference

The main objective of the problem is to find the label of the observation sequence. The probability of observation sequence $O = \{O_1, O_2,, O_T\}$ belonging to gesture g i.e $P(O|\lambda, g)$ can be computed using forward-backward recursion algorithm. Lets consider a forward variable for gesture g defined as

$$\alpha_t^{(g)}(i) = P(O_1, O_2,, O_t, q_t = S_i | \lambda, g) \tag{5}$$

the probability of observing sequence $O_1,, O_t$ and $q_t = S_i$ given the model parameters. We can recursively find $\alpha_t^{(g)}(i)$ using the following equations

$$\alpha_1^{(g)}(i) = \pi_i^{(g)} b_i(O_1), \quad 1 \leq i \leq N. \tag{6}$$

$$\alpha_t^{(g)}(j) = \left[\sum_{i=1}^N \alpha_{t-1}^{(g)}(i) a_{ij}^{(g)} \right] b_j(O_t), \quad 1 \leq j \leq N.$$

$$2 \leq t \leq T. \tag{7}$$

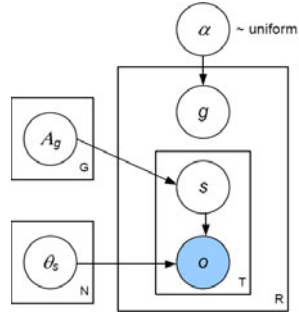


Fig. 1. Plates diagram for a shared state parameter model

$$P(O|\lambda, g) = \sum_{i=1}^N \alpha_T(i) \tag{8}$$

Similar to forward variable, lets define a backward variable for gesture g as

$$\beta_t(i)^{(g)} = P(O_{t+1}, O_{t+2}, \dots, O_T | q_t = S_i, \lambda, g) \tag{9}$$

is the probability of observation sequence O_{t+1} to O_T given the state at time $= S_i$ and λ . We can again solve for this variable using recursion as follows

$$\beta_T^{(g)}(i) = 1, \quad 1 \leq i \leq N. \tag{10}$$

$$\beta_t^{(g)}(i) = \sum_{j=1}^N a_{ij}^{(g)} b_j(O_{t+1}) \beta_{t+1}^{(g)}(j), \quad 1 \leq i \leq N. \tag{11}$$

$t = T - 1, T - 2, \dots, 1.$

From the above equations we can find single best state sequence $Q = q_1, q_2, \dots, q_T$ for a given observation sequence $O = O_1, O_2, \dots, O_T$ given the model parameters and gesture g using Viterbi algorithm.

$$P^{*(g)} = \max_{q_1, q_2, \dots, q_T} P(q_1, q_2, \dots, q_T, O_1, O_2, \dots, O_T | \lambda, g). \tag{12}$$

We assign an observation sequence a gesture label g which maximizes $\arg \max_g P^{*(g)}$.

3.2 Parameter Estimation

To estimate the maximum likelihood parameters of the model, we use the standard Baum-Welch iterative procedure, paying special attention to the shared states.

Let there be R observation sequences in the dataset with their corresponding labels. Let η be a variable such that

$$\eta(g, r_c) = \begin{cases} 1 & \text{if } g = r_c \\ 0 & \text{otherwise,} \end{cases} \tag{13}$$

where r_c is the label for sample r and $1 \leq r_c, g \leq G$.

1. Estimation of initial state probabilities

$$\overline{\pi_i^{(g)}} = \sum_{r=1}^R \eta(g, r_c) \left[\frac{\alpha_1^{(g)r}(i) \beta_1^{(g)r}(i)}{P_r^{(g)}} \right] \tag{14}$$

where $P_r^{(g)} = P(O^r | \lambda, g)$ is the probability of observation sample r given gesture g and the parameters.

2. Estimation of Transition probabilities

$$\overline{a_{ij}^{(g)}} = \frac{\sum_{r=1}^R \eta(g, r_c) \left[\frac{1}{P_r^{(g)}} \sum_{t=1}^{T_r-1} \alpha_t^{(g)r}(i) a_{ij}^{(g)} b_j(O_{t+1}^r) \beta_{t+1}^{(g)r}(j) \right]}{\sum_{r=1}^R \eta(g, r_c) \left[\frac{1}{P_r^{(g)}} \sum_{t=1}^{T_r} \alpha_t^{(g)r}(i) \beta_t^{(g)r}(i) \right]} \tag{15}$$

3. Estimation of observation density parameters

$$\overline{\mu_j} = \frac{\sum_{r=1}^R \frac{1}{P_r^{rc}} \sum_{t=1}^{T_r} \alpha_t^{rc}(j) \beta_t^{rc}(j) O_t^r}{\sum_{r=1}^R \frac{1}{P_r^{rc}} \sum_{t=1}^{T_r} \alpha_t^{rc}(j) \beta_t^{rc}(j)} \tag{16}$$

$$\overline{\Sigma_j} = \frac{\sum_{r=1}^R \frac{1}{P_r^{rc}} \sum_{t=1}^{T_r} \alpha_t^{rc}(j) \beta_t^{rc}(j) (O_t^r - \mu_j)(O_t^r - \mu_j)}{\sum_{r=1}^R \frac{1}{P_r^{rc}} \sum_{t=1}^{T_r} \alpha_t^{rc}(j) \beta_t^{rc}(j)} \tag{17}$$

3.3 Inference with Multiple Hand Locations

If we had just one observation for each frame, we could use the conventional Viterbi algorithm to calculate $P^{*(g)}$. But because we have multiple hand locations at each frame, we need to maximize one more term in the Viterbi algorithm which will give the best hand location sequence. It is important to note that we do not have multiple hand locations in the training data. In the training data users have worn colored gloves so that the hand location can be extracted easily and the model is trained on these hand locations. Since we cannot find exact hand location during testing, we find multiple hand location hypothesis for each frame. We present a decoding algorithm similar to Viterbi which will not only give the best state sequence but also gives the best hand location sequence h_1, h_2, \dots, h_T . To do this, we will define a quantity $\delta_t^{(g)}(i, k)$:

$$\delta_t^{(g)}(i, k) = \max_{\substack{q_1, q_2, \dots, q_{t-1} \\ h_1, h_2, \dots, h_{t-1}}} P(q_1, q_2, \dots, q_t = i, h_1, h_2, \dots, h_t = k, O_1, O_2, \dots, O_t | \lambda, g). \tag{18}$$

i.e., $\delta_t(i, k)$ is the best score along a single path at time t , which takes into account the first t observations and ends in state S_i and hand location H_k . We can recursively define $\delta_{t+1}(j, k)$ as

$$\delta_{t+1}^{(g)}(j, l) = [\max_i \max_k \delta_{t-1}(i, k) a_{ij}^{(g)}] b_j(O_{t+1, l}), 1 \leq i, j \leq N$$

$$1 \leq l, k \leq K. \tag{19}$$

To find the best state sequence and best hand location sequence we need to keep track of the arguments which maximized equation [19](#). We can do this using matrices $\psi_t(j)$ and $\varphi_t(l)$. The algorithm to find the best state sequence and hand location sequence can be found using the following procedure:

1. Initialization:

$$\delta_1^{(g)}(i, k) = \pi_i^{(g)} b_i(O_{1k}), \quad 1 \leq i \leq N. \quad (20)$$

$$\psi_1(i) = 0. \quad (21)$$

$$\varphi_1(k) = 0. \quad (22)$$

2. Recursion:

$$\delta_t^{(g)}(j, l) = [\max_i \max_k \delta_{t-1}(i, k) a_{ij}^{(g)}] b_j(O_{t+1,l}), 2 \leq t \leq T$$

$$1 \leq i, j \leq N.$$

$$1 \leq k, l \leq K. \quad (23)$$

$$\psi_t(j) = \arg \max_{1 \leq i \leq N} [\delta_{t-1}(i, k) a_{ij}^{(g)}], \quad 1 \leq k \leq K. \quad (24)$$

$$\varphi_t(l) = \arg \max_{1 \leq k \leq K} [\delta_{t-1}(i, k) a_{ij}^{(g)}], \quad 1 \leq i \leq N. \quad (25)$$

3. Termination:

$$P^{*(g)} = \max_{1 \leq i \leq N} \max_{1 \leq k \leq K} [\delta_T(i, k)] \quad (26)$$

$$q_T^{*(g)} = \arg \max_{1 \leq i \leq N} [\delta_T(i, k) a_{ij}^{(g)}], \quad 1 \leq k \leq K. \quad (27)$$

$$h_T^{*(g)} = \arg \max_{1 \leq k \leq K} [\delta_T(i, k) a_{ij}^{(g)}], \quad 1 \leq i \leq N. \quad (28)$$

4. state sequence and hand location sequence:

$$q_t^{*(g)} = \psi_{t+1}(q_{t+1}^{*(g)}), \quad t = T - 1, T - 2, \dots, 1. \quad (29)$$

$$h_t^{*(g)} = \varphi_{t+1}(h_{t+1}^{*(g)}), \quad t = T - 1, T - 2, \dots, 1. \quad (30)$$

4 Experiments and Results

The experiment involved having subjects create gestures by writing the ten Palm Graffiti Digits in the air, as shown in Figure 2. The goal of the experiment as conducted by Alon et al. in [1] was to evaluate the performance of gesture recognition systems. In our work, we extend this to also investigate the nature of sub-gestures. Each video clip depicts a user writing each of the ten digits in sequence. A total of 30 training and 30 testing sequences were produced for each of the gestures, resulting in a total of 600 sequences. Half was used for training and validation, and the other half for testing.

During training and validation, we varied the number of hidden states from 8 to 16 and found the optimal number of states to be 10. This is a coincidence and is not related to the fact that there are ten gestures being analyzed. We



Fig. 2. Palm’s Graffiti Digits on which users were asked to create gestures

Table 1. Recognition results (% accuracy)

Model	4D features	only optical flow features
Proposed Method	93.33	81.67
HMM (non shared)	88.8	48

Table 2. Hand location Results

Candidate locations	Accuracy
1	84.00
2	85.67
3	87.33
4	89.00
5	90.00
6	92.00
7	93.33
8	92.33
9	91.33
10	91.00

Table 3. Confusion matrix generated from classifying Palm Graffiti digits

0	1	2	3	4	5	6	7	8	9
0.7	0	0	0	0	0	0.2	0	0.07	0.03
0	0.8	0	0.03	0	0	0.1	0.07	0	0
0	0	0.97	0	0	0	0	0	0	0.03
0	0	0	1	0	0	0	0	0	0
0	0	0	0	0.97	0	0	0	0.03	0
0	0	0	0	0	0.97	0	0	0	0.03
0	0	0	0	0	0	1	0	0	0
0	0	0	0	0.03	0	0	0.94	0	0.03
0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	1

performed the same experiments using (i) our proposed shared parameter model using the complete 4D feature set (ii) the non-shared parameter model using the complete 4D feature set (iii) our proposed shared parameter model using only the temporal features and (iv) the non-shared parameter model using only the temporal features. The accuracy results are presented in Table 1. Table 2 shows how the accuracy of the proposed method varies with change in number of candidate hand locations.

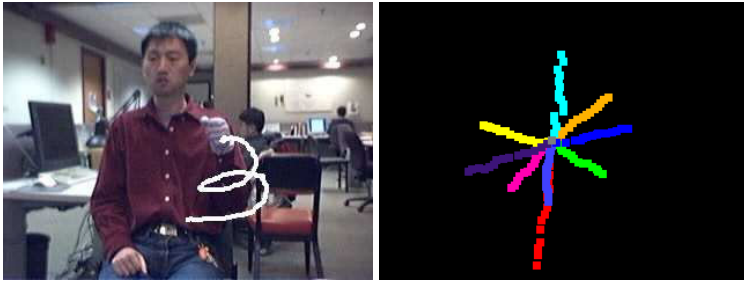


Fig. 3. Left image: a user performing the gesture for Palm's Graffiti Digit '3'; right image: the set of 10 sub-gestures learned by the shared parameters HMM, from training data. The static point in the middle of the image represents the "no-motion" sub-gesture.

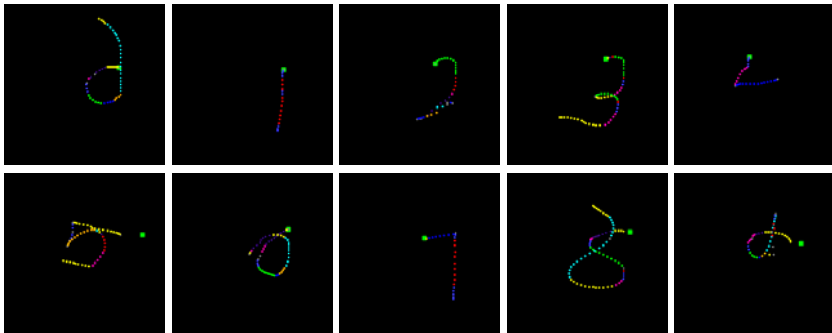


Fig. 4. Examples of labeling different observation sequences with their underlying shared sub-gesture states. In this illustration, the shown sequence for '0' was misclassified as '6' and also, the shown sequence for '4' was misclassified as '7'. All other classifications are correct. The color representation of the sub-gesture labels is shown in Figure 3 above.

To demonstrate the performance of our model, we performed gesture recognition on the dataset, classifying each gesture into one of the 10 digit classes. The overall accuracy was 93.3% and the resulting confusion matrix is shown in Figure 3.

5 Conclusion and Future Work

We have presented a computationally efficient framework for explicit gesture and implicit sub-gesture modeling. We have shown that there is an improvement in the accuracy rate when the HMM parameters are shared across gestures, resulting in a set of learned sub-gestures.

The main drawback of the proposed approach is inherited from the fact that it is generative in nature, and generative models in general are more error prone

than their discriminative counterparts. As a next step, we intend to compare the performance of our approach with the state-of-the-art discriminative models described in [10].

Although generative models such as ours generally require a large amount of training data, by sharing the parameters across multiple HMMs, we require less training data.

Going forward, we intend to extend our technique to gesture spotting and perform more testing on significantly larger datasets.

References

1. Alon, J., Athitsos, V., Yuan, Q., Sclaroff, S.: A unified framework for gesture recognition and spatiotemporal gesture segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31 (2008)
2. Assan, M., Grobel, K.: Video-based sign language recognition using hidden markov models. In: Wachsmuth, I., Fröhlich, M. (eds.) *GW 1997. LNCS (LNAI)*, vol. 1371, pp. 97–109. Springer, Heidelberg (1998)
3. Brand, M., Oliver, N., Pentland, A.: Coupled hidden markov models for complex action recognition, pp. 994–999 (1997)
4. Gavrilu, D.M., St. Runge, W.: *The visual analysis of human movement: A survey* (1999)
5. Jones, M.J., Rehg, J.M.: Statistical color models with application to skin detection. *International Journal of Computer Vision* 46, 81–96 (2002)
6. Kapoor, A., Picard, R.W.: A real-time head nod and shake detector. In: *PUI 2001: Proceedings of the 2001 workshop on Perceptive user interfaces*, pp. 1–5 (2001)
7. Kienzle, W., Bakir, G., Franz, M., Schlkopf, B.: Face detection - efficient and rank deficient 17, 673–680 (2005)
8. Li, H., Greenspan, M.: Multi-scale gesture recognition from time-varying contours. In: *ICCV 2005: Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV 2005)*, vol. 1, pp. 236–243. IEEE Computer Society, Washington, DC, USA (2005)
9. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: *Proceedings of Imaging Understanding Workshop*, vol. 2, pp. 121–130 (1981)
10. Morency, L.P., Quattoni, A., Darrell, T.: Latent-dynamic discriminative models for continuous gesture recognition. In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2007* (2007)
11. Pavlovic, V.I., Sharma, R., Huang, T.S.: Visual interpretation of hand gestures for human-computer interaction: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19, 677–695 (1997)
12. Sato, Y., Kobayashi, T.: Extension of hidden markov models to deal with multiple candidates of observations and its application to mobile-robot-oriented gesture recognition. In: *International Conference on Pattern Recognition*, vol. 2 (2002)
13. Starner, T., Pentland, A.: Real-time American Sign Language recognition from video using hidden Markov models. In: *International Symposium on Computer Vision*, p. 265 (1995)

14. Wang, S.B., Quattoni, A., Morency, L.P., Demirdjian, D., Darrell, T.: Hidden conditional random fields for gesture recognition. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 1521–1527 (2006)
15. Wu, Y., Huang, T.S.: Vision-based gesture recognition: A review. In: Braffort, A., Gibet, S., Teil, D., Gherbi, R., Richardson, J. (eds.) GW 1999. LNCS (LNAI), vol. 1739, pp. 103–115. Springer, Heidelberg (2000)
16. Yang, R., Sarkar, S.: Gesture recognition using hidden markov models from fragmented observations. In: CVPR 2006: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 766–773. IEEE Computer Society, Washington, DC, USA (2006)

Author Index

- Abdmouleh, Fatma 321
Abdullah, Rosni 248
Absil, P.-A. 358
Aggarwal, Jake K. 1
Alberola-López, Carlos 420
Alegre, Enrique 410
- Barucci, Elena 298, 372
Barczi, Krisztina 194
Battaglino, Daniela 275
Baudrier, Étienne 284
Ben Hamza, A. 84, 444
Benitez, Ania 9
Bersani, Marcello M. 222
Bhattacharya, Bhargab B. 70
Bhowmick, Partha 70
Biswas, Arindam 70
Brimkov, Valentin E. 5
Brocchi, Stefano 298, 372
Browet, Arnaud 358
- Cartade, Colin 157
Casaseca-de-la-Higuera, Pablo 420
Cerdán, Sebastián 9
Charrier, Emilie 132
Cherubini, Alessandra 222
China, Alejandro 469
Ciecholewski, Marcin 432
Cordero-Grande, Lucilio 420
Ćurić, Vladimir 385
- Damiand, Guillaume 56
Daurat, Alain 284, 321
de Albuquerque, Victor H.C. 456
Dupas, Alexandre 56
- Falcão, Alexandre X. 456
Frigeri, Achille 222
Frosini, Andrea 275, 298
- García-Olalla, Oscar 410
Geetha, H. 261
Gerard, Yan 144
Ghosh, Subarna 483
- González-Castro, Víctor 410
Gonzalez-Diaz, Rocio 107
Govindaraju, Venu 483
Guerra, Concettina 13
- Heyvaert, Michaël 182
- Jarray, Fethi 311
Joshi, Swapna 410
- Kalyani, T. 261
Kardos, Péter 17, 31
Karmakar, Nilanjana 70
Kerautret, Bertrand 398
Khader, Mohammed 444
- Lachaud, Jacques-Olivier 43, 56, 132,
208
Lago, Luis 9
Lalitha, D. 235
Lamar, Javier 107
Li, Chunyuan 84
Lindblad, Joakim 385
Lizarbe, Blanca 9
López-Larrubia, Pilar 9
Lukić, Tibor 335
- Mahalingam, Kalpana 248
Malgireddy, Manavender R. 483
Malgouyres, Rémy 157
Mercat, Christian 157
Mina, Marco 13
- Nagar, Atulya K. 248
Nagy, Benedek 194
Németh, Gábor 17
Nwogu, Ifeoma 483
- Palágyi, Kálmán 17, 31
Papa, João P. 456
Pereira, Clayton R. 456
Pont, Oriol 346
Prasanna Venkatesan, A.S. 261
Provençal, Xavier 208
Provot, Laurent 144

- Rangarajan, K. 235
Reulke, Ralf 168
Rinaldi, Simone 275
Roussillon, Tristan 43, 398
Rueß, Dominik 168
- Samieinia, Shiva 96
Samir, Chafik 157
Sanchez-Montañes, Manuel 9
Silva, Cleiton C. 456
Sladoje, Nataša 385
Slapal, Josef 120
Subramanian, K.G. 248
- Tajine, Mohamed 284, 321
Tavares, João Manuel R.S. 456
Thomas, D.G. 235, 261
Tlig, Ghassen 311
Turiel, Antonio 346
- Umble, Ronald 107
- Vacavant, Antoine 398
Van Dooren, Paul 358
Veelaert, Peter 182
Vegas-Sánchez-Ferrero, Gonzalo 420
- Yahia, Hussein 346