



UNIVERSITÄT
DES
SAARLANDES



mpi
max planck institut
informatik

TOPOLOGICAL ANALYSIS OF DISCRETE SCALAR DATA

DAVID GÜNTHER

DISSERTATION ZUR ERLANGUNG DES GRADES
DES DOKTORS DER INGENIEURWISSENSCHAFTEN
DER NATURWISSENSCHAFTLICH-TECHNISCHEN FAKULTÄTEN
DER UNIVERSITÄT DES SAARLANDES

SAARBRÜCKEN, 2012

Betreuer / Supervisor:

Dr. Tino Weinkauff
MPI Informatik, MPC-VCC, Saarbrücken, Germany

Eingereicht am / Thesis submitted:

24. September 2012 / September 24th, 2012

Gutachter / Reviewers:

Dr. Tino Weinkauff
MPI Informatik, MPC-VCC, Saarbrücken, Germany
Prof. Dr. Hans-Peter Seidel
MPI Informatik, Saarbrücken, Germany
Prof. Dr. Olga Sorkine
ETH Zurich, Switzerland
Prof. Eugene Zhang, Ph.D.
Oregon State University, Corvallis, USA

Datum des Kolloquiums / Date of Defense:

18. Dezember 2012 / December 18th, 2012

Dekan / Dean:

Prof. Dr. Mark Groves
Universität des Saarlandes, Saarbrücken, Germany

Vorsitzender / Chair:

Prof. Dr. Christian Theobaldt
Universität des Saarlandes, Saarbrücken, Germany

Prüfer / Examiners:

Dr. Tino Weinkauff
MPI Informatik, MPC-VCC, Saarbrücken, Germany
Prof. Dr. Hans-Peter Seidel
MPI Informatik, Saarbrücken, Germany
Prof. Dr. Olga Sorkine
ETH Zurich, Switzerland
Prof. Eugene Zhang, Ph.D.
Oregon State University, Corvallis, USA

Protokollant / Recorder:

Dr. Michael Wand
MPI Informatik, Saarbrücken, Germany

Max-Planck-Institut Informatik
Campus E1 4
66123 Saarbrücken
Germany

Eidesstattliche Versicherung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus anderen Quellen oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet. Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form in einem Verfahren zur Erlangung eines akademischen Grades vorgelegt.

David Günther
Saarbrücken, September 24, 2012

Abstract

This thesis presents a novel computational framework that allows for a robust extraction and quantification of the Morse-Smale complex of a scalar field given on a 2- or 3-dimensional manifold. The proposed framework is based on Forman's discrete Morse theory, which guarantees the topological consistency of the computed complex. Using a graph theoretical formulation of this theory, we present an algorithmic library that computes the Morse-Smale complex combinatorially with an optimal complexity of $O(n^2)$ and efficiently creates a multi-level representation of it. We explore the discrete nature of this complex, and relate it to the smooth counterpart. It is often necessary to estimate the feature strength of the individual components of the Morse-Smale complex – the critical points and separatrices. To do so, we propose a novel output-sensitive strategy to compute the persistence of the critical points. We also extend this well-founded concept to separatrices by introducing a novel measure of feature strength called separatrix persistence. We evaluate the applicability of our methods in a wide variety of application areas ranging from computer graphics to planetary science to computer and electron tomography.

Kurzzusammenfassung

In dieser Dissertation präsentieren wir ein neues System zur robusten Berechnung des Morse-Smale Komplexes auf 2- oder 3-dimensionalen Mannigfaltigkeiten. Das vorgestellte System basiert auf Forman's diskreter Morsetheorie und garantiert damit die topologische Konsistenz des berechneten Komplexes. Basierend auf einer graphentheoretischer Formulierung präsentieren wir eine Bibliothek von Algorithmen, die es erlaubt, den Morse-Smale Komplex mit einer optimalen Komplexität von $O(n^2)$ kombinatorisch zu berechnen und effizient eine mehrskalige Repräsentation davon erstellt. Wir untersuchen die diskrete Natur dieses Komplexes und vergleichen ihn zu seinem kontinuierlichen Gegenstück. Es ist häufig notwendig, die Merkmalsstärke einzelner Bestandteile des Komplexes – der kritischen Punkte und Separatrizen – abzuschätzen. Hierfür stellen wir eine neue outputsensitive Strategie vor, um die Persistenz von kritischen Punkten zu berechnen. Wir erweitern dieses fundierte Konzept auf Separatrizen durch die Einführung des Wichtigkeitsmaßes Separatrixpersistenz. Wir evaluieren die Anwendbarkeit unserer Methoden anhand vielfältiger Anwendungen aus den Gebieten der Computergrafik, Planetologie, Computer- und Elektronentomographie.

Summary

The rapid increase of the amount of image data produced in industry and scientific research requires the availability of efficient tools to analyze this data. Various features in the data have image densities larger or smaller than their neighborhood; they are the extremal structures of the data. For instance, valleys on a planet's surface are given as minimal lines in an elevation map, while membranes of cells can be described as surfaces of minimal intensity in an electron tomogram.

A super-set of these extremal structures is the Morse-Smale complex defined by the underlying data. The Morse-Smale complex consists of critical points – the local minimal, saddle and maximal points – and separatrices which are integral lines of the gradient connecting the critical points. There are currently two main areas of research in this field: solving algorithmic challenges in the construction of topological data structures, and applying the topological techniques to extract meaningful features in specific applications.

In this thesis, we address both areas: We develop an algorithmic library that allows for an efficient computation of the Morse-Smale complex. We propose strategies how the feature strength of each component of the Morse-Smale complex can be estimated. We evaluate our tools in different application scenarios. In particular, this thesis targets the following topics:

Morse-Smale Complex. Typically, the Morse-Smale complex is computed numerically using the information given by the gradient and the Hessian of the image. However, a robust numerical treatment can be very challenging, since the data is usually affected by noise. In this dissertation, we develop combinatorial algorithms based on discrete Morse theory that allow for an efficient, consistent and robust computation of the Morse-Smale complex for 2- and 3-dimensional scalar data. In contrast to previous approaches, our Morse-Smale complex computation has a provably optimal complexity of $O(n^2)$ with n denoting the size of the input.

Persistent Homology. The evolution of topological features at consecutive thresholds is described by persistent homology. This concept has drawn much attention since it robustly identifies the most dominant topological features of the data. In particular, pairs of critical points are assigned an importance measure which allows to separate spurious critical points from dominant ones. The computation of persistence is usually done algebraically. However, such an approach suffers from huge memory consumption. In this thesis, we alleviate this effect by reducing the size of the processed data. The Morse-Smale complex allows for an efficient computation of persistent homology since it is, in general, much smaller than the input data but still contains all necessary information. Using the Morse-Smale complex, we can significantly reduce the memory

consumption for large 3-dimensional data (up to a factor of 30 in real-world data). Additionally and in contrast to the classic approach, the computational complexity of our persistence computation scheme is output-sensitive, i.e., it depends only on the number of critical points and not on the size of the data.

Hierarchy of Morse-Smale Complexes. Usually, different levels of detail of the Morse-Smale complex are of interest in order to separate small-scale and large-scale structures. This is especially important if the image data is affected by noise or sampling artifacts. A distinction of the noise-induced structures and large-scale structures can be obtained using the concept of persistent homology. Persistence robustly pairs critical points according to their presence in the data. While this pairing gives rise to a natural hierarchy in two dimensions, it does not directly yield a hierarchy in three dimensions, in general. In particular, the construction of a perfect hierarchy becomes NP-hard in higher dimensions, and heuristics must therefore be involved. In this thesis, we present algorithms that efficiently create a hierarchy of Morse-Smale complexes for 2- and 3-dimensional data such that the hierarchy reflects the global appearance of the topological structures.

The Persistence of a Separatrix. In various applications the feature strength of an individual separatrix or parts thereof needs to be estimated. The above hierarchy is insufficient for this task since it estimates the strength of each separatrix by a constant value. Within this dissertation, we propose a novel measure of feature strength to assess the importance of separatrices. This measure is globally defined and based on persistent homology. The strength of a separatrix is not assessed by a constant value but by an interval of importance estimating each point of a separatrix individually. Not only does it allow to determine the most important (parts of) separatrices, it also serves as a robust filtering measure of noise-induced structures.

Applicability and Evaluation. The methods presented in this thesis allow for an efficient computation of the extremal structures in the data. We explore the usefulness of our computational framework in a wide variety of application areas: computer graphics, planetary science, fluid dynamics, molecular and cell biology, computer tomography. To understand the strengths and limitations of our proposed framework, we evaluate it to techniques that are frequently used in data analysis and discuss the differences and similarities between them.

Danksagung

Diese Arbeit hat einen langen Weg hinter sich, und ohne die Unterstützung von Freunden, Kollegen und Familie wäre sie wohl nicht möglich gewesen. Ihr Beitrag lässt sich kaum beziffern, dennoch möchte ich im folgenden versuchen meine Dankbarkeit ihnen gegenüber auszudrücken.

Zuerst möchte ich mich bei meinem Doktorvater und Freund Tino Weinkauff bedanken. Ohne ihn hätte ich das Gebiet topologische Datenanalyse nie kennengelernt und nie erfahren welches Potential in dieser Theorie steckt. Er zeigte mir, wie man seine eigenen Ideen verfeinert, formuliert, und praktisch umsetzt. Ohne seine Unterstützung, sein Vertrauen und seine Anleitung in den letzten Jahren würde ich nicht da sein, wo ich heute bin. Dafür werde ich dir immer dankbar sein.

Ich möchte außerdem Hans-Peter Seidel danken. Als Direktor des MPI Informatik hat er mir die Möglichkeit gegeben, meine Doktorarbeit hier nach meinen Vorstellungen fertigzustellen. Ich bin zudem sehr dankbar, dass er sich bereiterklärt hat, diese Arbeit zu begutachten. Vielen Dank.

Ich möchte mich ganz herzlich bei Olga Sorkine bedanken. Ihr promptes Einverständnis meine Doktorarbeit zu begutachten hat mich zutiefst erfreut. Vielen Dank für dein ausführliches und detailliertes Gutachten.

I also would like to thank Eugene Zhang for his thorough review. It was a pleasure discussing with you during your time at the MPI. I'm very grateful that you could make it to come to my defense.

In nächtelangen Diskussionen mit Jan Reininghaus ist der Grundstein für diese Arbeit entstanden. Diese Diskussionen waren sehr kontrovers, aber auch immer sehr unterhaltsam. Vielen Dank dafür und für alles wofür hier kein Platz ist.

Ich möchte mich zudem bei Wolfgang Baumeister, Reiner Hegerl, und Alexander Rigort für ihre Unterstützung und ihr Vertrauen bedanken. Am MPI Biochemie habe ich mich immer wie zu Hause gefühlt. Ich möchte mich auch bei einer Vielzahl von Leuten aus Berlin bedanken: Conny Auer, Daniel Baum, Ingrid Hotz, Jens Kasten, Michael Koppitz, Andrea Kratz, Falko Marquardt, Jan Sahner. Ohne euch hätte die Arbeit nicht annähernd so viel Spass gemacht. Ganz besonders möchte ich dabei meiner Büro-WG-Mitbewohnerin Uli danken. Danke für alles! Es war eine großartige Zeit.

Ein spezieller Dank geht an Cathleen Krambeer. Vielen Dank für deine langjährige Unterstützung. Ohne dich würde es diese Arbeit nicht geben. Außerdem möchte ich Daniel Scherzer danken. In der Endphase der Arbeit hat er immer für die nötige Ablenkung gesorgt.

Zum Schluss möchte ich mich bei meinen Eltern und meiner Schwester bedanken. Ihr habt mir den Rückhalt gegeben, diese Arbeit zu schreiben. Vielen Dank für eure grenzenlose Geduld, euer Vertrauen und Verständnis, und eure Unterstützung.

Contents

1	Introduction	13
1.1	Motivation	13
1.2	Related Work on Topological Data Analysis	15
1.3	Overview of the Thesis	19
2	From Smooth to Discrete Morse Theory	21
2.1	Cell Complexes and Homology Groups	22
2.2	Smooth Morse Theory	26
2.3	Algorithmic Challenges of Smooth Morse Theory	30
2.4	Discrete Morse Theory	31
3	Computational Discrete Morse Theory	35
3.1	Preliminaries	36
3.2	The Graph Theoretical Setting	38
3.2.1	The Cell Graph	38
3.2.2	Morse Matchings	40
3.2.3	Critical Points and Combinatorial Separation Lines	41
3.2.4	Properties of Combinatorial Separation Lines	42
3.3	Computation of Combinatorial Gradient Fields	45
3.4	Computation of Morse-Smale Complexes	47
3.4.1	The Algorithm by Robins et al.	47
3.4.2	An Optimal Algorithm	48
3.4.3	Computational Complexity	50
3.4.4	Implementational Details	51
3.4.5	Performance Comparison	53
3.5	A Hierarchy of Combinatorial Gradient Fields	54
3.5.1	Simplification in 2 Dimensions	56
3.5.2	2D Examples	58
3.5.3	Simplification in 3 Dimensions	59
3.5.4	Performance Analysis	63
3.5.5	Properties of the Hierarchization	65
3.5.6	3D Examples	68
4	Quantification of Critical Points	71
4.1	Persistent Homology	73
4.2	Related Work on Persistent Homology	74
4.3	Computation of Persistent Homology	75
4.3.1	3D Examples	75

4.3.2	Discussion	77
4.4	Topological Simplification and Persistence	80
4.5	Application: Feature Point Correspondences	82
4.5.1	Related Work on Shape Matching	82
4.5.2	The Matching Pipeline	84
4.5.3	Applications	86
4.5.4	Experiments	87
4.5.5	Discussion	91
5	Quantification of Separatrices	93
5.1	Separatrices and Height Ridges	94
5.2	The Persistence of a Separatrix	96
5.2.1	The 2-dimensional Case	97
5.2.2	The 3-dimensional Case	99
5.2.3	Method Overview	102
5.3	Applications of Separatrix Persistence	103
5.3.1	Salient Edges on Surfaces	104
5.3.2	Valleys on the Martian Surface	108
5.3.3	Computer Tomography	109
5.3.4	Scalar Quantities in Fluid Dynamics	111
5.3.5	Cell Biology	114
6	Discussion	115
6.1	Smooth vs. Discrete Topological Structures	115
6.1.1	The Steepest Descent	117
6.1.2	Probabilistic Steepest Descent	118
6.1.3	Empirical Convergence	119
6.2	The Importance of Separatrices	124
6.3	Extension to Higher Dimensions	125
6.4	From Discrete to Smooth	125
7	Conclusions and Future Work	127
	Bibliography	128
	List of Figures	142
	List of Tables	150
	Index	152

Chapter 1

Introduction

1.1 Motivation

Measurements and numerical simulations are essential to get a deeper understanding of the underlying processes in many scientific and industrial areas.

The objects under study within the data can be very complex. For perception of these objects, the data must therefore be visualized. Common approaches visualize the data by filtering and mapping techniques. For example, the color range and transparency in a rendering system is adapted to the data to present it to the user in the form of 2- or 3-dimensional images or animations. For some purposes, this qualitative form of visualization is sufficient. However, for a deeper analysis of the structural features in the complex settings of the application specific landscapes, it is necessary to characterize them in terms of parameters such as surface areas, volumes, distances or angles. A quantitative analysis of this kind makes it necessary to identify and localize the features of interest and to separate them from the background.

This processing could be performed manually under visual control. However, the more and more automated data acquisition methods result in a huge amount of data sets. Their manual processing is therefore a very time-consuming and cost-intensive task. This processing also entails a user-specific bias. Some features might be overlooked or falsely interpreted. An objective extraction of the features of interest is a challenging task for a user and cannot always be given.

Computer-assisted feature extraction methods are therefore frequently employed in many applications. Even though the results of a computer-based technique might not be perfect when compared with the ground truth – which is usually unknown – it is reproducible and allows for a meaningful comparison of the results from a multitude of experiments. To assure the quality of the results, such methods should have certain properties to be useful in practice: Firstly, they should be as automated as possible to enable an objective analysis and thereby avoiding a user-specific point of view. Secondly, the methods should be stable and reliable to guarantee that their output can be compared and further processed. Thirdly, they should be efficient such that automated analysis is less time-consuming than manual processing. Given such a technique, image-based features can be robustly extracted in an objective manner.

A property of image-based features is that they have an extremal attribute. Based on this attribute, we can classify them in three categories: Point-like features describe local peaks in the data such as local temperature maxima in a heat-exchanger simulation or

local minima in a pressure field derived from a fluid dynamics simulation. Line-like features such as valleys in an elevation map or blood vessels in a computer-tomogram appear as lines of maximal or minimal intensity in the data. In volumetric data, surface-like features such as the membrane of a cell in an electron tomogram can be found as a surface of minimal intensity.

Two different concepts are commonly used to extract such features: local analysis or analysis based on the global point of view.

The local analysis investigates local properties of the feature's intensity distribution. For instance, a template describing a representative distribution of the feature [Fra06] can be used to trace line-like structures [RGH⁺12b]. Such a technique solely uses the pixel information of the image. However, the design of such a template is a very application-specific task and depending on the complexity of the template the analysis can be very computationally expensive.

A more general approach is the use of information based on the derivatives of an image. This enables a convexity/concavity analysis of a local neighborhood [KvD93]. Based on such an analysis, ridges and valleys can be extracted from an image. This idea goes back to De Saint-Venant [dSV52] in 1852, and is nowadays a frequently used technique in image analysis. A recent variant of this idea are Height Ridges [Ebe96] which investigate the eigenvalues and -vectors of the underlying Hessian. Depending on the sign of the eigenvalues, k -dimensional ridges are extracted from the image. Such local analysis is very sensitive to noise in the image data; the level of noise gets strongly amplified in the image derivatives. These distortions challenge a robust estimation of the eigenvalues and -vectors, and thereby the computation of the ridges.

The global analysis investigates the monotony behavior within the image. The watershed approach is herein a commonly used technique. This theory has its roots in the work of Maxwell [Max70] in 1870, and was generalized in 1934 in Morse's seminal work *Calculus of Variations in the Large* [Mor34]. The image decomposes into its ascending and descending manifolds that originate at its critical points. In each of these manifolds, the image behaves monotonically and a break of the monotony only occurs across their borders. Later on, Smale [Sma61b] extended this theory allowing to define the Morse-Smale complex: Critical points are in a neighborhood relation and connected by unique integral lines. An important property of the Morse-Smale complex is its structural stability, i.e., small changes within the image do not change the relationship of the critical points.

An advantage of such global analysis is its robustness due to theoretical constraints. The global point of view puts features into a relationship. Morse theory links the occurrence of critical points of a function to the topology of the underlying domain. For example, this constraint enforces that between two minima of a function a maximum must occur. Such information is not available in a local analysis, and a misclassification can very well occur here.

Although the constraints increase the robustness in theory, it can be difficult to enforce them in the algorithmic design. Numerical algorithms that compute the Morse-Smale complex usually perform local analysis by investigating each item individually. The global constraints are not directly employed, which might also result in certain misclassifications and an inconsistent Morse-Smale complex. Recently, a first strategy was proposed by Chattopadhyay [Cha11] to compute this complex numerically in a certified manner using interval arithmetic. Although the domain is currently restricted to be planar or an implicit surface, the computation of the Morse-Smale complex overcomes the typical numerical difficulties. However, the approach is limited to analytic functions, and therefore not suited for the scope of this thesis.

Robin Forman translated in his work *Morse Theory for Cell Complexes* [For98b] the idea of Morse theory from the smooth into a discrete setting. Here, it is no longer assumed that a sufficiently smooth function is given, only values assigned to the elements of the discretized domain are necessary. The coupling of the critical points of a function to the topology of the discretized domain is formulated in a purely combinatorial manner. In some sense, this theory provides a discretization of the set of admissible Morse-Smale complexes for a given domain. Only a finite number of possibilities exists, and, hence, this version of Morse theory is very suited for the algorithmic design. Since statements in discrete Morse theory are formulated combinatorially, algorithms do not need to make use of derivatives or any numerical procedure.

Discrete Morse theory not only allows for a stable and reliable computation of the Morse-Smale complex, it also enables a multi-level representation of this complex. While this is theoretically possible in the smooth setting [Mil65a] as well, a consistent numerical realization is very challenging. On the other hand, the combinatorial setting of Forman's theory directly allows for it.

In the course of this thesis, we extract image-based features as a subset of the discrete Morse-Smale complex. The proposed algorithmic framework enjoys the global point of view of the Morse-Smale complex. Due to the combinatorial setting, we can directly include the global constraints of Morse theory, and it enables the design of robust algorithms that compute image-based features without any computational parameters. Since no user-interaction is necessary, features are extracted solely from the information in the image, and therefore in an objective manner. The user is left with a hierarchy of features and only needs to choose an appropriate level of detail for further statistical analysis. The input is represented in a graph-theoretical setting that allows for a running-time and memory efficient algorithmic design. Thanks to this design, large 2- and 3-dimensional images can be processed on commodity hardware.

1.2 Related Work on Topological Data Analysis

In the following, we give a brief overview about topological data analysis. We focus here on computational topology [EH10], due to its ability to extract relevant features of the analyzed data.

From the topological point of view, features very often correspond to changes of isocontours. A common way to detect these changes is by the use of contour trees [Mor66] or the more general Reeb graphs [Ree46]. Both approaches compute a graph which tracks the evolution of connected components of a scalar function defined on a manifold over an increasing isolevel. While the contour tree requires that the data is given on a simply connected domain, the Reeb graph works for arbitrary domain topology. The computed graph consists of nodes, which represent the births, splits and merges of the components, and arcs describing the adjacency of the components.

In the last decades, simple, fast and robust algorithms have been proposed [TV98, CSA00, CMEH⁺04, TGSP09], which made them very useful in different areas of application: clean isosurface extraction [CSvdP04] or feature driven visualization metaphors [WBP07].

However, the contour tree and the Reeb graph do not capture all topological information: Not all genus changes of the isocontours are detected by them [EH10]. This is especially crucial in case of volumetric data. Encapsulated cavities cannot be detected by Reeb graphs. A representation of the topological structure that captures all changes is the Morse-Smale complex [Mor34, Sma61b]. In contrast to the Reeb graph, this

complex describes the topological structure by analyzing the gradient flow behavior of the given data [Mil63]. In this work, we use the Morse-Smale complex to analyze the data since all topological information is represented by it.

The Morse-Smale complex consists of critical points and separatrices. In this setting, the critical points are the local minimum-, saddle-, and maximum points, while the separatrices are special gradient lines connecting the critical points [Cay59]. The Morse-Smale complex induces a segmentation of the data into regions of monotonic behavior [Mil65b] and is strongly related [NS94] to the concept of the watershed transform [Max70]. In fact, the separatrices form a super-set of the watersheds and water-courses [GC95, LLSV99].

There are three established methods to compute the Morse-Smale complex. The classic approach employs numerical methods. In this setting, the critical points are given by computing all zeros of the gradient. While the zeros can be computed exactly in the piecewise linear context, a Newton-Raphson scheme [New69, Rap90] or similar needs to be applied, in general. The separatrices are extracted by solving a system of autonomous ordinary differential equations. Starting at a saddle point, one follows the gradient in the direction of the eigenvectors of the Hessian [Wei08]. Using interval methods, the 2-dimensional Morse-Smale complex can be extracted in a numerically certified manner [CVY12] on planar domains or implicit surfaces.

The second approach works in a piecewise linear context. In this setting, the critical points are given by an analysis of the lower star of each vertex [Ban70]. Similar to the classic approach, Banchoff's definition also allows for higher-order critical points. The separatrices are typically approximated as a sequence of steepest edges in the triangulation. Edelsbrunner et al. [EHZ03, EHNP03] proposed the first approach for the 2- and 3-dimensional case. In this approach, the separatrices follow the edges of the triangulated domain allowing them also to merge. Bremer et al. [BEHP04] refined this approach by subdividing the triangulation in the vicinity of the separatrices. Due to the subdivision, the separatrices do not need to merge and could follow the interpolated gradient.

In this work, we build upon a purely combinatorial approach proposed by Robin Forman [For98b, For01] to compute the Morse-Smale complex. Such an approach lends itself to computational purposes due to its discrete nature [Lew05, Gyu08, Bau11]. In contrast to the classic approach, it computes the Morse-Smale complex directly on the grid given by the data. In this setting, the critical points are defined by the topological changes in the sub-level sets of the data [Mil63]. These topological changes can be computed efficiently by constructing a combinatorial gradient [RWS11]. In contrast to the classic and piecewise linear approach, this combinatorial setting only allows for first-order critical points. The relationship of the critical points to the piecewise linear context was discussed by Lewiner [Lew12]. The separatrices are computed by starting at the (combinatorial) saddle points and following the grid along the combinatorial gradient field.

The first computational realization of Forman's theory was presented by Lewiner et al. [LLT03, Lew05] to compute the homology groups of 2- and 3-dimensional manifolds. In this framework, a consistent combinatorial gradient field is computed, and the Morse-Smale complex is implicitly defined therein. The combinatorial gradient field is represented by hypergraphs and hyperforests, which allow for a very compact and memory efficient representation of the Morse-Smale complex. However, the framework is only applicable to relatively small 3-dimensional data sets since links in the graph are traversed multiple times during the gradient field construction. This results in infeasible running time for data sets of reasonable size.

Robins et al. [RWS11] followed the idea of Lewiner [Lew05]. They use a classic breadth-first search in their combinatorial gradient field to compute the Morse-Smale complex. However, the multiple graph traversals result in cubic running-time, making this approach infeasible for real-world data. Inspired by the work of Lewiner and Robins et al., we also use breadth-first searches in our graph, but we avoid multiple traversals. Our approach results in quadratic running-time, which is provably the optimal computational complexity.

An alternative approach to extract the essential critical points and separation lines is proposed by Gyulassy et al. [GNP⁺06, GNPH07, Gyu08]. Their main idea is to construct a gradient-like field based on a watershed-like transform and extract the critical points and separation lines by a field traversal. This results in a graph structure that connects the critical points by the separation lines. The proposed technique allows for fast streaming of very large data. However, no guarantees about the consistency of the graph with respect to the domain topology are given.

A Morse decomposition could also be computed using Conley index theory [Con78]. This theory was applied by Chen et al. [CMLZ08] and Szymczak et al. [SZ12] in the 2-dimensional discrete vector field context. An alternative approach was proposed by Chen et al. [CDS⁺12] by introducing piecewise constant vector fields, combining the Conley index theory with differential inclusions [AC84]. Although the scalar field context is much more restrictive than the vector field context, i.e., no closed streamlines can occur, the ideas are applicable to scalar fields as well, as shown by Chen et al. [CDS⁺12]. An algorithmic generalization to higher dimensions, however, is still open.

Sampling artifacts and noise may create spurious topological structures. In many applications, it is therefore beneficial to separate these structures from the dominant ones representing the large-scale behavior of the analyzed data. The removal of topological structures such as critical points was first investigated by Smale [Sma61a] and Milnor [Mil65a]. Recently, several techniques from the more general area of vector field analysis were proposed for simplification [dLvL99a, dLvL99b, TSH00, TRS03a, TRS03b, WTS⁺05]. The continuous setting of the presented ideas challenges their application to discrete data. However, they can be applied in the smooth scalar field context. Edelsbrunner et al. [EHZ03] presented a simplification strategy for the piecewise linear case in two dimensions. In their approach, the algorithmic challenge is the unfolding of higher-order critical points.

In this work, we make use of the concept of persistent homology [EH08], which is an algebraic method for measuring the importance of critical points. The basic concepts have been independently developed by Frosini and Landi [FL99], Robins [Rob00], and Edelsbrunner et al. [ELZ02]. Persistent homology was originally introduced for piecewise-linear data. Bauer [Bau11] translated this concept into the discrete setting of discrete Morse theory which enables the persistence computation also for combinatorial critical points.

A strong stability result for the persistence measure has been proven by Cohen et al. [CSEH07]. This result guarantees that persistence can be used as a signature. Whenever two persistence outputs are different, we know that the functions are definitely different. It has also been shown that persistence can be used to simplify the Morse-Smale complex [Zom01] of a 2-dimensional scalar function defined on a smooth manifold. However, this result does not generally hold in higher dimensions as discussed by Bauer [Bau11].

A heuristic that is commonly used to simplify the Morse-Smale complex is the height difference of adjacent critical points. For the smooth 2-dimensional case, this

heuristic coincides with persistence [DLL⁺10]. However, this is no longer true in higher dimensions [Bau11].

In Lewiner’s approach [Lew05], a hierarchy of Morse-Smale complexes is implicitly given by a hierarchy of hyperforests. However, the hierarchy of hyperforests suffers from multiple graph traversals. The simplification of the topological graph of Gyulassy [Gyu08] is more intricate. This graph is explicitly represented and neighborhood information and geometric embeddings need explicit storage. In three dimensions, certain pairs of critical points can be connected arbitrarily often [TWHS03a, TWHS07]. This results in a quadratic amount of memory [GBPH11] to store this information. Gyulassy et al. [GBPH11] address this problem by changing the ordering of cancellations. They introduce a heuristic to defer a cancellation if the number of arcs in the graph that were newly generated exceeds a user-defined threshold. The choice of a practical threshold depends on the data and might result in an insufficient number of levels of detail.

In this work, we also use the heuristic of the height difference to guide the hierarchy construction. However, the memory consumption and running-time of our algorithms behave well in practice even for large 3-dimensional data.

Given the relevant topological features, they can be used to process and analyze the underlying objects in several aspects: Since scalar fields usually suffer from noise, the relevant topological features can be used to denoise the data in a constrained manner, i.e., the most dominant topological structures should also be present in the denoised version [GZ07, WGS10, Bau11, JWS12].

The resolution of surfaces and grids can be adapted to their topological features. For example, Theisel et al. [TRS03b] and Lewiner et al. [LLT04] proposed a mesh compression technique while Dong et al. [DBG⁺06] used the topological information to optimize quadrangular meshes.

The design of transfer functions to visualize 3-dimensional scalar data by volume rendering heavily depends on the complexity of the data. Using the relevant topological features, the design can be guided by them to allow for convincing visualizations [FAT99, WS02, WDC⁺07, KKH12].

An interesting application in the area of image processing was proposed by Chen et al. [CFL11]. They used topological information such as the number of connected components or connectivity to constrain image segmentation.

The analysis of time-dependent data is a challenging subject since topological features such as critical points need to be followed over time. Theisel et al. [TS03] and Weinkauff et al. [WTGP11] proposed an algorithmic framework for the tracking of critical points in the smooth setting. A combinatorial variant thereof was proposed by Reininghaus et al. [RKWH12] based on the work of King et al. [KKM05]. Kasten et al. [KRHH11, KHNH12] used this combinatorial framework to compute vortex regions and their merge graph based on the acceleration magnitude of a vector field.

The interested reader is referred to the work of López et al. [LLSV99] for an overview and evaluation of image-based feature extraction. For a more detailed survey on persistent homology, we recommend the work of Edelsbrunner et al. [EH08]. An overview of the extraction of topological structures in the smooth setting is given in the survey of Laramée et al. [LHZP07].

1.3 Overview of the Thesis

In the following, we give an overview of the structure of this thesis. The thesis introduces and describes a computational framework for the analysis of 2- and 3-dimensional scalar data using topological methods. The techniques presented herein are applicable to a variety of application data. The kind of data ranges from cryo-electron tomograms showing macro-biological structures within cells to elevation maps of the Martian surface.

This thesis consists of two parts. The first part deals with the algorithmic and technical aspects of the framework. In Chapters 2 and 3, we develop the algorithms needed for the topological data analysis. We discuss their properties, analyze their complexity, and illustrate them in experiments. The second part focuses on the computation of the feature strength of topological structures. In Chapters 4 and 5, we present strategies for efficiently estimating the importance of critical points and separatrices. This assessment allows for a removal of those parts that carry no essential information.

We begin with the computation of the Morse-Smale complex and its multi-level representation. We first introduce in Chapter 2 the mathematical theory that forms the foundation of the algorithmic framework. We focus on the main statements of algebraic topology and Morse theory needed for the algorithmic design. However, not all details of these theories are given since this would go beyond the scope of this work. The interested reader is referred to standard textbooks presenting the theory. The aim of Chapter 2 is rather to convey the ideas of Morse theory and its implications on topological data analysis.

The algorithmic framework is presented in Chapter 3. The framework should be thought of as a library providing different algorithms that can be included in other systems. The design of this library was guided by the following ideas:

Efficiency. The algorithms should be as efficient as possible to be useful in practice. This concerns the computational complexity, the running times, and the memory consumption.

Simplicity. The overall structure and operations done within the algorithms should be as simple as possible to allow for a straight-forward implementation.

Usability. There should be no user-interaction necessary to run the algorithms. Therefore, the framework should be free of computational parameters.

With these ideas in mind, our framework targets three algorithmic challenges:

Data Structures. The framework is placed in a graph theoretical setting. This graph structure enables simple and efficient graph traversals. Depending on the input, the graph is explicitly (e.g., triangulated surface) or implicitly (e.g., uniform grid) represented. Additionally, this graph structure allows for efficient set operations. For example, subsets of links of the graph can be easily intersected.

Morse-Smale Complex. In contrast to previous algorithms, our framework allows for an optimal computation of the Morse-Smale complex in three dimensions with a complexity of $O(cn)$ with c denoting the number of critical points and n the size of the input. Previous techniques needed $O(n^3)$.

Multi-level representation. A hierarchy of 2- and 3-dimensional Morse-Smale complexes can be constructed within the framework. This hierarchy is implicitly

given in a sequence of combinatorial gradient flows, and allows for a distinction of spurious and important topological structures. In contrast to previous approaches, the relative memory consumption is constant *and* the running times behave almost linearly for well-defined data.

The algorithms within the library need no computational parameters. The computation of the Morse-Smale complex and its multi-level representation is solely based on the scalar values of the input, allowing an objective analysis of the data. The algorithms of Chapter 3 are based on the works [RGH⁺10, GRP⁺11, GRWH11, GRWH12, RGH⁺12a].

The second part of the thesis describes the quantification of topological structures, i.e., measuring the feature strength of the critical points, separation lines and surfaces. We begin in Chapter 4 with an output-sensitive algorithm for computing persistent homology. Given this strategy, we develop a new algorithmic pipeline to find feature-point correspondences. In this chapter, we make the following contributions:

Persistent Homology. Assuming that the Morse-Smale complex is given, we present a strategy that allows to compute persistent homology with a complexity of $O(c^3)$ with c denoting the number of critical points of the input. In contrast to the algebraic approaches, the complexity of this strategy is output-sensitive since it only depends on the number of critical points and not on the size of the data.

Feature Point Correspondences. Persistent homology extracts the essential feature points. We use this strategy to match persistent feature points between two near-isometric surfaces. Our novel algorithmic pipeline is presented in the second part of Chapter 4.

In Chapter 5, we extend the concept of persistence to separation lines and surfaces. This defines a novel importance measure for higher dimensional topological features that allows to distinguish their important parts. In this chapter, we make the following contributions:

Separatrix Persistence. We introduce a novel importance measure for separation lines and surfaces in the 2- and 3-dimensional case based on the concept of persistent homology. The multi-level representation of the Morse-Smale complex (Chapter 3) does not respect the change of importance along a topological structure, in contrast to our new importance measure. With this measure, we can remove unimportant parts of these structures.

Applications and Comparison. Given the information provided by separatrix persistence, we use it to extract the extremal structures within 2- and 3-dimensional scalar data. We provide different use-cases to illustrate the robustness and applicability of our new importance measure in real-world data. This includes a thorough comparison to local analysis techniques.

The strategies presented in the Chapters 4 and 5 are based on the works [WG09, GMW⁺10, GRWH11, GRWH12, YGW⁺12, GSW12, PGW12].

We conclude this thesis with a discussion about the applicability of the computational framework. Especially, we investigate the effect of the discrete nature of the input data on the geometric embedding of the topological structures. We also discuss possible limitations and extensions in the area of topological data analysis.

Chapter 2

From Smooth to Discrete Morse Theory

In this chapter, we present the theoretical foundation for the algorithms and techniques developed in this work. These concepts not only guarantee the correctness but also the robustness and reliability of the algorithms.

The domain where the input function is defined on is usually given as a manifold. Such manifolds can be decomposed into cells, which are the main building blocks for the theory. We therefore start with the definition of the cell-complex [Whi49]. Using this cell-complex, we can define the Betti-numbers allowing a first differentiation of the input data based only on the topology of the underlying manifold. The aim of Section 2.1 is to give a brief informal overview about the needed concepts in algebraic topology. We introduce notation and recapitulate the main statements of algebraic topology needed for the algorithmic design. We follow here the elementary book by Hatcher [Hat02], which provides a complete introduction to this topic as well as the proofs of the presented theorems. The reader should be familiar with the notion and the basics of simplicial or cubical topology. A nice introduction to these topics is given in the books by Edelsbrunner and Harer [EH10] and Kaczynski et al. [KMM04].

Having introduced the basic definitions, we will concentrate on the concept of smooth Morse theory [Mor34] in Section 2.2. This theory relates the occurrence of critical points of a smooth scalar function to the topology of the underlying manifold. The focus will be the Morse-Smale complex [Sma61b] and its relationship to the underlying manifold given by the Morse inequalities as well as the Morse isomorphism. We will follow here the expositions of Milnor [Mil63, Mil65a] and refer the interested reader to it for a complete description and the proofs.

Morse-Smale complexes are usually computed using numerical algorithms and procedures [Wei08, Cha11]. Before we introduce a combinatorial alternative, we discuss the numerical challenges of such algorithms in Section 2.3.

In Section 2.4, we introduce the theoretical concept of the seminal work of Robin Forman [For98a, For01] who translated statements of Morse theory from the smooth setting into a discrete one. This discrete setting allows for combinatorial algorithms that are free of any numerical issues, which greatly improves their robustness. We concentrate on the main theorem of discrete Morse theory. The technical aspects of this theory are presented in Chapter 3. For the complete proof of the main theorem, we refer to Forman [For98a]. We only provide a sketch of it.

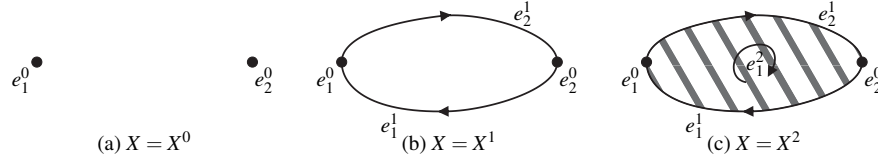


Figure 2.1: Examples of cell complexes.

2.1 Cell Complexes and Homology Groups

The input function $f : \Omega \rightarrow \mathbb{R}$ is usually defined on a manifold-like domain $\Omega \subset \mathbb{R}^d$. Typical examples that arise in practice are spheres, tori, planes, or full cubes embedded in \mathbb{R}^3 . Such domains have specific properties. They encapsulate cavities or contain tunnels. Some of them can also be retracted to a single point. These properties constrain the scalar functions defined on them. To understand these functions, we first need to understand how such manifolds are assembled. The manifolds decompose into cells, which are k -dimensional disks, and attaching maps. This composition is called the cell-complex X and it is a compact representation of the manifold Ω . We now define this complex and investigate what kind of information we can extract from it.

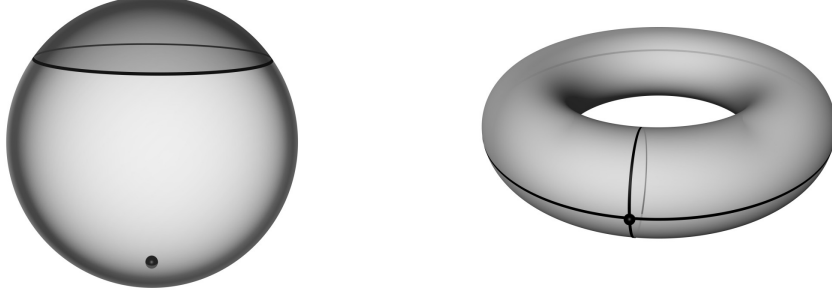
Before we give the formal definitions of the cells and the cell complex, we want to motivate the idea by a simple example: We begin with two vertices as shown in Figure 2.1a. Such vertices are interpreted as 0-dimensional cells. We connect these two vertices by two edges as shown in Figure 2.1b. The edges are 1-dimensional cells and their start- and end-points are attached to the vertices. We have constructed an ellipse. In the third step (Figure 2.1c), we fill the ellipse by a 2-dimensional stretched disk such that the boundary of that disk is attached to the two edges. We could now continue to attach further cells to the filled ellipse. However, the procedure is always the same: given a new cell, we attach it to the existing complex by gluing its boundary to the cells of the complex. The dimension of the complex is given by the largest dimension of the contained cells. This sketches the idea of a cell complex. We now give its formal definition.

The building blocks of a *cell-complex* X are cells homeomorphic to open disks and attaching maps identifying their boundary. The cell-complex X itself is given by induction. Let e^k be a k -dimensional open disk with boundary ∂e^k (which is a $k-1$ dimensional sphere S^{k-1}). We start with a discrete set of points X^0 , which are 0-dimensional cells. The k -skeleton X^k is given by attaching k -dimensional cells e_α^k to X^{k-1} by continuous maps $\varphi_\alpha : S^{k-1} \rightarrow X^{k-1}$. We can interpret X^k as the quotient space of the disjoint union $X^{k-1} \sqcup_\varphi e_\alpha^k$ with the equivalence relation $x \sim \varphi_\alpha(x)$ for $x \in \partial e_\alpha^k$. The equivalence relation represents the gluing of the boundary ∂e_α^k of a disk e_α^k to the X^{k-1} -skeleton. The maps φ_α define a natural inclusion of the skeletons $X^0 \subset X^1 \subset X^2 \subset \dots$ called the *CW-decomposition* of X .

In case of $X = X^d$, the cell complex X is called finite dimensional of dimension $d < \infty$. The manifolds shown in Figure 2.2 are all finite dimensional. For example, the sphere (Figure 2.2a) is constructed by attaching a 2-cell to a 0-cell¹ while the torus (Figure 2.2b) is constructed using a single 0-cell, two 1-cells and one 2-cell. We only consider finite dimensional complexes in this work.

The attaching of higher dimensional cells defines a neighborhood relationship. If

¹The 1-dimensional boundary of the 2-sphere is attached to the single 0-cell by a constant mapping. No 1-cell is necessary for this attachment.



(a) Cell decomposition of a sphere: The boundary (black line) of a 2-cell (dark gray surface) is attached to a single 0-cell (black vertex).

(b) Cell decomposition of a torus: The boundary of two 1-cells (black lines) is attached to a single 0-cell (black vertex). The shell of the torus is represented by a 2-cell (light gray surface).

Figure 2.2: Examples of cell decompositions of smooth manifolds.

e^{k+1} is attached to e^k , we write $e^k < e^{k+1}$ and call e^k *face* of e^{k+1} and e^{k+1} *coface* of e^k . The cells $\{e^k\}$ form a basis of the free abelian group C_k . The elements of C_k are called *chains* and are given as a formal sum $\sum n_\alpha e_\alpha^k$ with $n_\alpha \in \mathbb{Z}$. Considering the oriented cell complex shown in Figure 2.1b, elements of C_1 are $e_1^1 + e_2^1$ and $-e_2^1$, for example. For the former element, the chain starts at e_2^0 , traverses e_1^1 following its orientation, passes e_1^0 , and follows e_2^1 until it reaches its start point e_2^0 again. The latter chain, however, consists only of e_2^1 but traversed in opposite direction.

Coefficients $n_\alpha \in \mathbb{Z}$ indicate how often and in which direction a cell in a given chain of C_k is passed. In this work, however, we do not take the orientation of a cell complex into account. Therefore, it suffices to use coefficients that only flag cells belonging to a specific chain. The coefficient group $\mathbb{Z}_2 = \mathbb{Z}/2\mathbb{Z}$ does exactly this. Every coefficient is taken modulo 2. Coefficients $n_\alpha \in \mathbb{Z}_2$ correspond to a specific group structure of C_k . The addition of chains is performed by taking the symmetric difference of their cells.

Given the chain groups C_k , we can now define a homomorphism $\partial_k : C_k \rightarrow C_{k-1}$ that maps each chain $c_k \in C_k$ to its boundary chain $\partial c_k \in C_{k-1}$. Noting that $\partial_{k-1} \partial_k = 0$, these homomorphism induce a *chain complex* C

$$C : C_k \xrightarrow{\partial_k} C_{k-1} \xrightarrow{\partial_{k-1}} \dots \xrightarrow{\partial_2} C_1 \xrightarrow{\partial_1} C_0 \xrightarrow{\partial_0} 0. \quad (2.1)$$

If we specify a unique index for each cell e^k in C_k , a k -chain corresponds to a vector in $\mathbb{Z}_2^{n_k}$, where n_k is the number of k -dimensional cells in the complex. The k -dimensional boundary operator ∂_k can be written as an $n_k \times n_{k-1}$ binary matrix (also denoted ∂_k) whose columns are the boundaries of the k -cells. Consider for example the boundary operators ∂_1 and ∂_2 of the cell complex shown in Figure 2.1c. The complex describes a full disk and consists of a single 2-cell ($n_2 = 1$), two 1-cells ($n_1 = 2$) and two 0-cells ($n_0 = 2$). Since we use coefficients in \mathbb{Z}_2 , the boundary matrices ∂_1 and ∂_2 are given by

$$\partial_1 = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \quad \text{and} \quad \partial_2 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

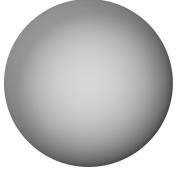


Name		Betti Numbers
Sphere		$\beta_0 = 1, \beta_1 = 0, \beta_2 = 1$
Torus		$\beta_0 = 1, \beta_1 = 2, \beta_2 = 1$
Klein Bottle		$(\mathbb{Z}): \beta_0 = 1, \beta_1 = 1, \beta_2 = 0$ $(\mathbb{Z}_2): \beta_0 = 1, \beta_1 = 2, \beta_2 = 1$

Table 2.1: Examples of homologically different complexes.

Consider now the chain $c = e_1^2$. Applying the composition $\partial_1 \partial_2(c)$ yields

$$\partial_1 \partial_2(c) = \partial_1 \left(\begin{pmatrix} 1 \\ 1 \end{pmatrix} \begin{pmatrix} 1 \end{pmatrix} \right) = \partial_1 \left(\begin{pmatrix} 1 \\ 1 \end{pmatrix} \right) = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

Note that we are in \mathbb{Z}_2 and every coefficient is taken modulo 2.

The elements of the image $B_k = \text{im } \partial_k$ of ∂_k are called the *boundaries* and the elements of the kernel $Z_k = \ker \partial_k$ of ∂_k the *cycles* of C_k . The image B_k and the kernel Z_k form a group with a structure induced by C_k . Since the composition $\partial_{k-1} \partial_k$ vanishes, there holds $B_k \subset Z_{k-1}$, and we can define the *homology group* as the quotient group

$$H_k(X) = Z_k / B_{k+1}. \quad (2.2)$$

The elements of the homology group H_k are those k -cycles that are not boundaries of any $k+1$ -chain. The homology groups give us the opportunity to topologically characterize a given complex X . In many applications, we are interested in the number of connected components or in the number of tunnels present in X . Such information are given by the elements of H_k . For example, the tunnel of a torus is represented by a 1-cycle, i.e., a sequence of edges that forms a ring and is not the boundary of any 2-chain. The *rank* of the homology group H_k is denoted by

$$\beta_k(X) = \text{rk } H_k(X) = \text{rk } Z_k - \text{rk } B_{k+1}. \quad (2.3)$$

β_k is also called the *k-th Betti number* of X and allows for a differentiation of cell-complexes: β_0 represents the number of components, β_1 the number of tunnels, β_2 the number of cavities encapsulated in a given complex X , etc..

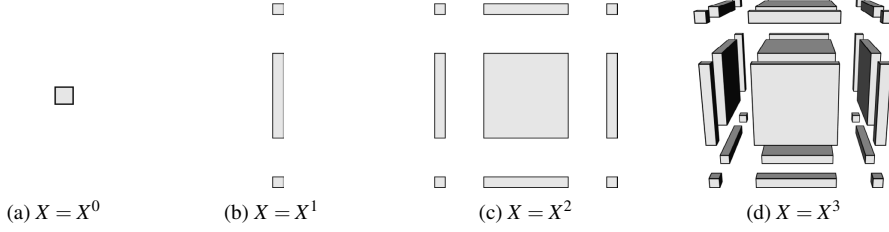


Figure 2.3: Examples of cubical complexes.

Figure 2.1 shows some examples with the corresponding Betti-numbers. We want to stress that something is lost using coefficients in \mathbb{Z}_2 . Complexes that contain torsion as the Klein Bottle are not fully described by \mathbb{Z}_2 , see the universal coefficient theorem [Hat02]. The Betti-numbers using \mathbb{Z}_2 are typically higher than using coefficients in \mathbb{Z} . In this case, for example, we cannot distinguish the Klein Bottle from a torus using the Betti numbers. In this work, however, all manifolds are orientable and therefore torsion-free. In this case, the homology groups are independent of the choice of coefficients. The Betti-numbers not only allow us to differentiate complexes by their homology, they are also the foundations for the analysis of a scalar function defined on this complex. In Section 2.2, we will see how the critical points of a scalar function relate to the Betti numbers.

In the following, we emphasize two special kinds of cell complexes that arise quite often, in practice. Surfaces and tetrahedral grids are usually given in a triangulated form. This triangulation induces a *simplicial complex* [EH10]. The elements of this complex are the k -simplices, which are given as the smallest convex set in \mathbb{R}^d which contains $k + 1$ vertices that do not lie in a k -dimensional hyperplane. The intersection of two k -simplices is thereby a $k - 1$ -simplex and also contained in the triangulation, or empty. An example of a simplicial complex is given on the right side.

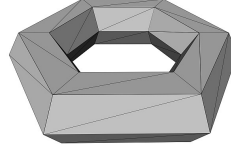


Image data are usually given on a *cubical complex* [KMM04]. The cells of this complex are vertices, edges, squares and full cubes. The vertices are defined as cross products of degenerated elementary intervals $[\ell, \ell]$ while the higher dimensional cells are given by the cross products of elementary intervals $(\ell, \ell + 1)$. The boundary operator ∂_k maps a k -dimensional cube to its $k - 1$ -dimensional boundary. For example, the boundary of a single full cube in \mathbb{R}^3 consists of six 2-dimensional squares while a square has four edges in its boundary each of them bounded by two vertices. Figure 2.3 shows examples of cubical cell complexes.

The class of manifolds that we consider in this work are orientable surfaces embedded in \mathbb{R}^3 and 2- and 3-dimensional cubes defined by a regular grid. Examples of such grids that arise in practice are: uniform-, rectilinear-, or curvilinear grids. Hence, we assume that a simplicial or cubical complex is given.

2.2 Smooth Morse Theory

In the following, we consider a sufficiently smooth scalar function $f : \Omega \rightarrow \mathbb{R}$ on a smooth manifold $\Omega \subset \mathbb{R}^d$. The *gradient* ∇f and the *Hessian* Hf of f are defined by

$$\nabla f = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_d} \end{pmatrix} \quad \text{and} \quad Hf = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1 \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_d} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_d \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_d \partial x_d} \end{pmatrix} \quad (2.4)$$

for a given coordinate system (x_1, x_2, \dots, x_d) and $\partial/\partial x_i$ denoting the partial derivative with respect to x_i . A point $p \in \Omega$ is called *critical* iff $\nabla f(p) = 0$ for any coordinate system. p is a *non-degenerated* critical point if additionally holds: $\det(Hf(p)) \neq 0$. A smooth function f that only contains non-degenerated critical points is called *Morse function* [Mor34]. The *index* of a critical point p is given by the number of negative eigenvalues of $Hf(p)$. p is called a *maximum* if it is of index d and a *minimum* if the index is 0. Otherwise, we call p a *saddle*.

Theorem 1 (Morse Lemma) *Let f be a Morse function and p be a critical point. There exists a local coordinate system (x_1, x_2, \dots, x_d) in a neighborhood $U(p) \subset \Omega$ such that $x_i(p) = 0$ for all i and f is locally given by*

$$f(q) = f(p) \pm x_1^2 \cdots \pm x_d^2 \quad (2.5)$$

for each point $q \in U(p)$.

From Theorem 1 follows that the critical points of a Morse function f are isolated, and, hence, finite in number. The index of a critical point corresponds to the number of negative signs in Equation 2.5.

There is a close relationship of the critical points to the homology of the lower level-sets of f . The *lower level set* of f is given by $L_r(f) = \{x \in \Omega \mid f(x) \leq r\}$ with isovalue $r \in \mathbb{R}$. Considering a sequence of monotonically increasing isovalues, the topological structure of $L_r(f)$ changes: connected components are born, merge, or create holes. This evolution is reflected by the Betti-numbers. Let us consider a function f given on a 2-dimensional manifold as shown in Figure 2.4. A connected component is always born at a minimum of f (Figures 2.4a and 2.4b). Given such an event, β_0 is increased by the number of new components. At a saddle, two situations can occur: either two connected components merge (Figure 2.4c), which decreases β_0 , or a component touches itself and forms a tunnel (Figure 2.4d), which increases β_1 . At a maximum, a tunnel is closed (Figures 2.4e and 2.4f), which results in a lowering of β_1 .

Let C^k be the set of critical points of index k . The above observation is formally given by the weak and strong Morse inequalities:

Theorem 2 (Weak Morse Inequalities) *Let f be a Morse function, then there holds for each k*

$$|C^k| \geq \beta_k(\Omega) \quad (2.6)$$

where $|C^k|$ denotes the cardinality of the set C^k and β_k the k -th Betti number.

The k -th Betti number is always a lower bound for the number of critical points of index k . For example, a Morse function f defined on a sphere must always contain a minimum and a maximum.

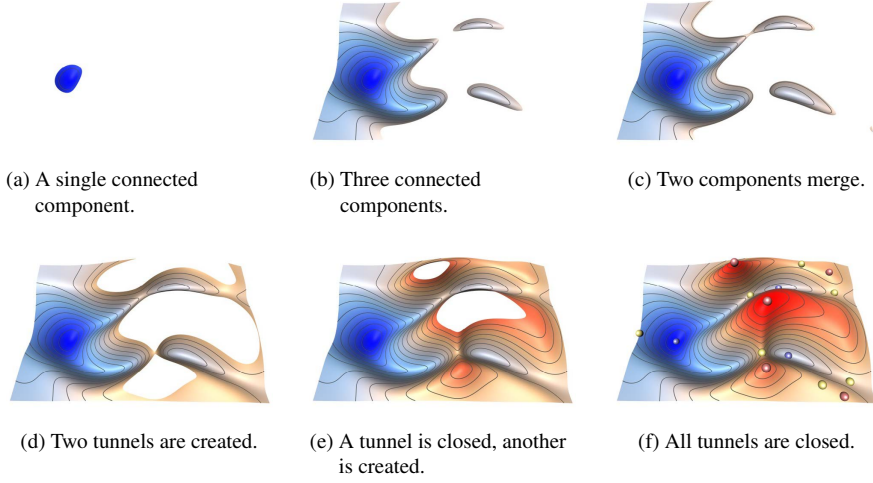


Figure 2.4: The lower level set $L_r(f)$ for different choices of r of an artificial height function f . The isovalue r increases from (a) to (f). The critical points of f are depicted in (f) as colored spheres: minimum (blue), saddle (yellow), maximum (red).

Theorem 3 (Strong Morse Inequalities) *Let f be a Morse function, then there holds for each k*

$$|C^k| - |C^{k-1}| + \dots \pm |C^0| \geq \beta_k(\Omega) - \beta_{k-1}(\Omega) + \dots \pm \beta_0(\Omega) \quad (2.7)$$

where $|C^k|$ denotes the cardinality of the set C^k and β_k the k -th Betti number.

Theorem 3 relates the occurrence of critical points. For example, if a Morse function f given on a sphere consists of two minima and a maximum, it must also contain a saddle point. Otherwise, the strong Morse inequalities (2.7) would be violated.

Besides these occurrence constraints, there is also a neighborhood relation between the critical points. The critical points are also part of a cell-complex that completely describes the behavior of a Morse function f : the Morse-Smale complex.

The Morse-Smale complex induces a decomposition of the domain Ω in regions where f behaves monotonically. It consists of critical points and integral lines of the gradient ∇f that connect pairs of critical points.

Let $\varphi : \mathbb{R} \times \Omega \rightarrow \Omega$ denote the negative gradient flow of f given by

$$\frac{\partial}{\partial t} \varphi(t, x) = -\nabla f(\varphi(t, x)) \quad \text{and} \quad \varphi(0, \cdot) = id_{\Omega}. \quad (2.8)$$

Given a non-degenerated critical point p of f , we define its *ascending* and *descending manifolds* by

$$\mathcal{A}_p = \{q \in \Omega \mid \lim_{t \rightarrow \infty} \varphi(t, q) = p\} \quad \text{and} \quad \mathcal{D}_p = \{q \in \Omega \mid \lim_{t \rightarrow -\infty} \varphi(t, q) = p\}. \quad (2.9)$$

The ascending and descending manifolds are herein embedded open discs of dimension

$$\dim(\mathcal{D}_p) = \text{index}(p) \quad \text{and} \quad \dim(\mathcal{A}_p) = \dim(\Omega) - \text{index}(p). \quad (2.10)$$

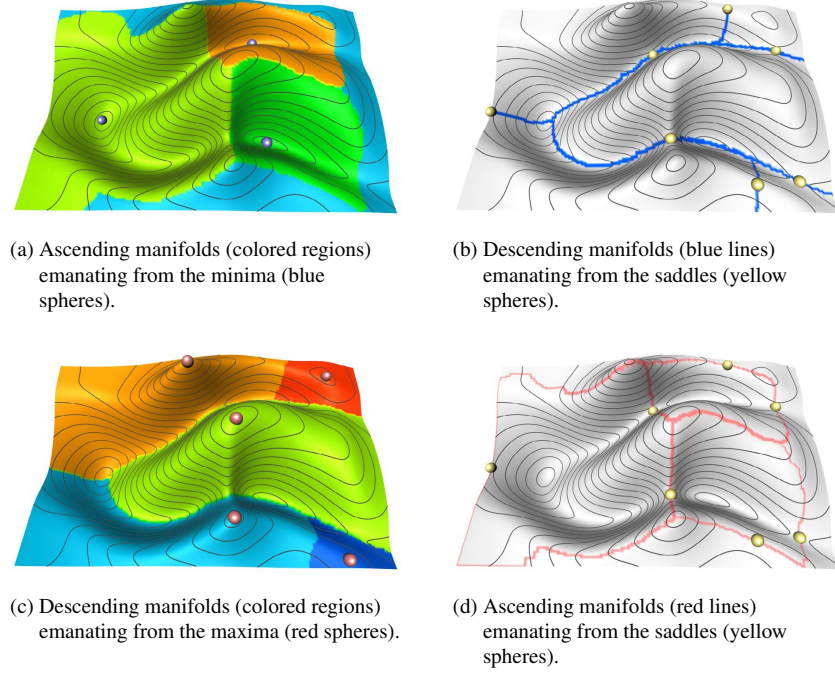


Figure 2.5: The ascending and descending manifolds of a terrain. The isolines are depicted as gray lines.

Figure 2.5 illustrate the sets \mathcal{D}_p and \mathcal{A}_p for the artificial height function shown in Figure 2.4. The ascending and descending manifolds of the minima and maxima, respectively, are 2-dimensional manifolds while the manifolds emanating from the saddles are 1-dimensional.

Given a pair of critical points (p, \tilde{p}) of f , we can now consider the intersection of \mathcal{A}_p and $\mathcal{D}_{\tilde{p}}$. We say the Morse function f fulfills the *Morse-Smale condition* [Sma61b] if the intersection $\mathcal{D}_{\tilde{p}} \cap \mathcal{A}_p$ is transversal for each pair of critical points (p, \tilde{p}) . The dimension of $\mathcal{D}_{\tilde{p}} \cap \mathcal{A}_p$ is then given by the indices of the involved critical points

$$\dim(\mathcal{D}_{\tilde{p}} \cap \mathcal{A}_p) = \dim(\mathcal{D}_{\tilde{p}}) + \dim(\mathcal{A}_p) - \dim(\Omega) = \text{index}(\tilde{p}) - \text{index}(p). \quad (2.11)$$

For example, two curves on a surface are transversal if they only intersect in points, while two surfaces are transversal if they intersect each other in curves. For a Morse function f fulfilling the Morse-Smale condition, the intersection of its ascending and descending manifolds define the Morse-Smale complex. Note that the Morse-Smale condition is not very restrictive. Every Morse function can be slightly perturbed such that it fulfills this condition without introducing new critical points.

An illustration for a simple 3-dimensional input function is given in Figure 2.6. We show here only the manifolds emanating from the saddles. The manifolds emanating from the extrema have similar characteristics as in the 2-dimensional case. Since the intersection of the 2-dimensional ascending and descending manifolds is transversal, saddles of opposite type are connected by 1-dimensional separatrices. For their numerical treatment, we refer to Theisel et al. [TWS03b].

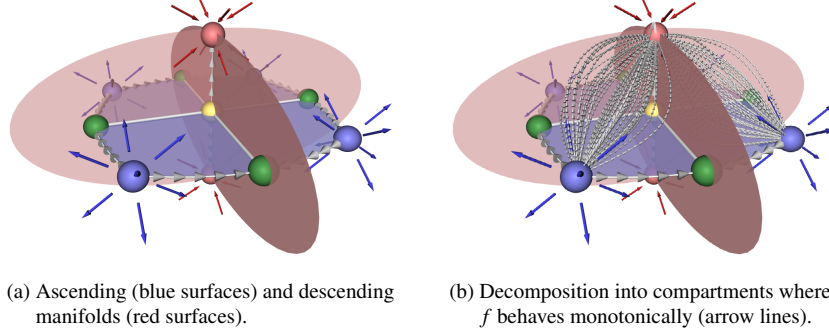


Figure 2.6: Illustration of the ascending and descending manifolds for a 3-dimensional input function. Arrows indicate the gradient flow. Blue and red spheres depict the minima and maxima while the saddles of index 1 and 2 are shown as green and yellow spheres, respectively. The 1-dimensional connections between two saddles are depicted as white lines. The ascending and descending manifolds decompose the domain into monotone regions.

The intersection of the ascending and descending manifolds consists of integral lines $\gamma: \mathbb{R} \rightarrow \Omega$ connecting pairs of critical points (p, \tilde{p}) given by

$$\gamma'(t) = -\nabla f(\gamma(t)), \quad \lim_{t \rightarrow -\infty} \gamma(t) = p \quad \text{and} \quad \lim_{t \rightarrow \infty} \gamma(t) = \tilde{p}. \quad (2.12)$$

We want to emphasize that the intersection may consist of a family of integral lines. Depending on the indices of the involved critical points, this family can span a surface- or volumetric-like object.

In the following, we only consider pairs of critical points (p, \tilde{p}) that are successive, i.e., $\text{index}(\tilde{p}) - \text{index}(p) = 1$. In this case, the integral lines defined by (2.12) are called *separatrices*. The separatrices induce a boundary operator $\partial: C^k \rightarrow C^{k-1}$ between the critical points with the property $\partial_{k-1} \partial_k = 0$ [Bot88, Sch93]. This operator defines a chain complex: *the Morse-Smale complex*

$$C^M: C^k \xrightarrow{\partial_k} C^{k-1} \xrightarrow{\partial_{k-1}} \dots \xrightarrow{\partial_2} C^1 \xrightarrow{\partial_1} C^0 \xrightarrow{\partial_0} 0. \quad (2.13)$$

For notational simplicity, we slightly abused the notion of C^k in Equation (2.13). Here, C^k denotes the free abelian group with coefficients in \mathbb{Z}_2 generated by the critical points of index k . Similar to (2.1), Equation (2.13) gives rise to homology groups $H_k^M(f)$. The most fundamental theorem in Morse theory states that the homology groups of the Morse-Smale complex carry the same information as the homology groups induced by the cell complex of a smooth manifold Ω .

Theorem 4 (Morse Homology) *Let Ω be a smooth compact manifold and $f: \Omega \rightarrow \mathbb{R}$ a Morse function fulfilling the Morse-Smale condition. Then there exists a canonical isomorphism*

$$H_k(\Omega) \simeq H_k^M(f). \quad (2.14)$$

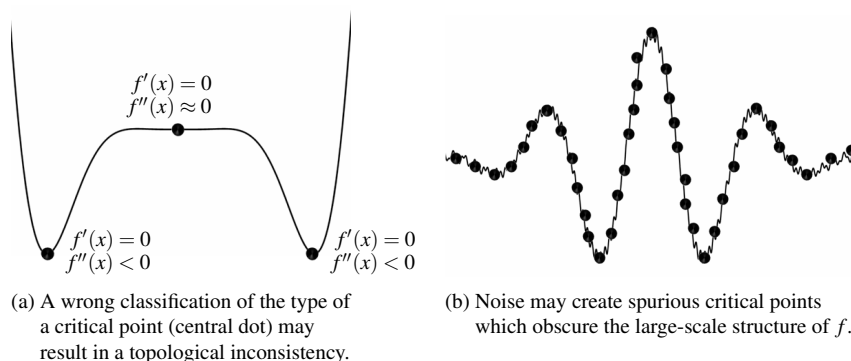


Figure 2.7: Challenges of numerical algorithms.

Note that Ω is usually equipped with a Riemann metric. However, Theorem 4 does not depend on it.

Due to Theorem 4, we can interpret the Morse-Smale complex as a very compact representation of an input function where all redundant information are compressed, and only the essential information – the critical points and their neighborhood relation – is present.

2.3 Algorithmic Challenges of Smooth Morse Theory

The computation of the Morse-Smale complex for a given function f can be very challenging. As described in Section 2.2, the critical points are computed by finding all zeros of the gradient and classifying them into minima, saddles, and maxima by the eigenvalues of the Hessian. The respective eigenvectors can be used to compute the separatrices as the solution of a system of autonomous ordinary differential equations.

One of the biggest challenges that such numerical algorithms face is the discrete nature of the critical points and their separatrices which necessitates many binary decisions. For example, the type of a critical point depends on the sign of the eigenvalues as illustrated in Figure 2.7a. From a global point of view, the central critical point must be a maximum. Using numerical algorithms, however, its classification is a purely local decision based on the eigenvalues. Since the neighborhood of critical point in the center is almost constant, a numerical algorithm may wrongly classify it as a minimum. Hence, the resulting Morse-Smale complex depends strongly on the computational parameters and numerical procedures.

From a topological point of view this can be quite problematic. As we have seen, Morse theory relates the Morse-Smale complex of a generic function to the topology of the underlying manifold. The topology of the manifold therefore restricts the set of admissible Morse-Smale complexes. For example, every scalar function defined on a sphere contains at least one minimum and one maximum. A first step to a certified numerical computation of Morse-Smale complexes for the special case of a planar 2-dimensional manifold was done by Chattopadhyay et al. [CVY12]. This work may serve as a reference for numerical algorithms, however, its extension to more complex manifolds and higher dimensions is still open.

In the next section, we introduce the basic ideas of Robin Formans seminal work on

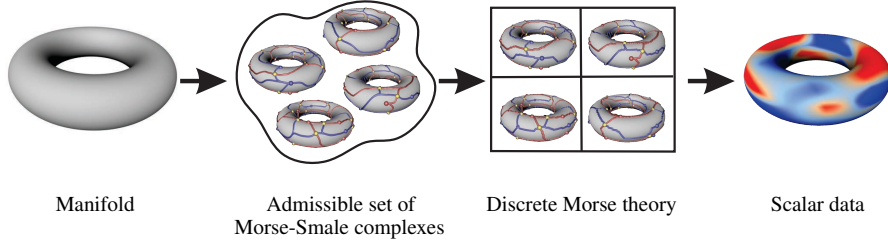


Figure 2.8: The topology of the domain restricts the set of admissible Morse-Smale complexes. Discrete Morse theory provides a discretization of this set. The objective of this theory is to find a complex from this finite set of Morse-Smale complexes that represents the given scalar data well.

discrete Morse theory. This theoretical framework provides a discretization of the infinite set of admissible Morse-Smale complexes. This combinatorial and discrete setting will allow us to design efficient and reliable algorithms to compute the Morse-Smale complex on 2- and 3-dimensional manifolds. The guaranteed topological consistency greatly improves the robustness of the presented algorithms. In some sense, it has built-in error correction: a single misclassification of a critical point cannot occur, as this would result in an inadmissible Morse-Smale complex.

A second advantage of the combinatorial setting is that it enables to handle spurious critical points. In many applications, one is interested in the large scale structure of the input data. However, noise arising in the data acquisition process may obscure it as illustrated in Figure 2.7b. An efficient analysis of the data is then almost impossible. One is therefore interested in a consistent and controllable removal of spurious critical points. Although this is theoretically also possible in the smooth setting [Mil65a], the algorithmic realization becomes much easier in the combinatorial world, since topological consistency is guaranteed.

An illustration of the idea of discrete Morse theory is given in Figure 2.8. The focus of this work lies in the last part of the shown pipeline: the efficient computation of the Morse-Smale complex and its quantification for a discrete scalar field given on a simplicial or cubical complex.

2.4 Discrete Morse Theory

Discrete Morse theory aims to describe an input function f in a combinatorial fashion. In Section 2.2, the input function f was a smooth Morse function given on a smooth manifold such that all needed derivatives exists. In the following, the function f only assigns a single scalar value to each cell of the complex X . We therefore interpret f as a discrete function. We will see that all of the statements from Section 2.2 can also be made if no continuity of f is required. Although the theoretical setting is applicable to general cell-complexes, we restrict ourselves in this section to simplicial complexes, for simplicity. The elements of X are the k -dimensional simplices.

The function $f : X \rightarrow \mathbb{R}$ is called *discrete Morse function* if for each simplex $\alpha^k \in X$ holds

$$\begin{aligned}
 |\{\beta^{k+1} \in X \mid \beta^{k+1} > \alpha^k, f(\beta^{k+1}) \leq f(\alpha^k)\}| &\leq 1 \quad \text{and} \\
 |\{\gamma^{k-1} \in X \mid \gamma^{k-1} < \alpha^k, f(\gamma^{k-1}) \geq f(\alpha^k)\}| &\leq 1.
 \end{aligned} \tag{2.15}$$

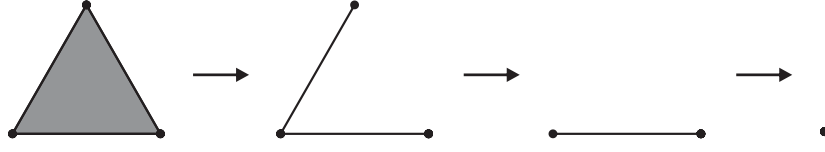


Figure 2.9: Simplicial collapse of a triangle to a vertex.

Loosely speaking, a discrete Morse function f assigns higher scalar values only to higher dimensional simplices with at most one exception, locally, at each simplex. The most trivial example of a Morse function is given by assigning each simplex its dimension. In the next chapter, we will see how real-world data such as images are transformed into a discrete Morse function.

If all (co)faces are of greater/smaller scalar value, the simplex α^k is called a critical simplex. More formally, the simplex $\alpha^k \in X$ is a *critical simplex* if

$$\begin{aligned} |\{\beta^{k+1} \in X \mid \beta^{k+1} > \alpha^k, f(\beta^{k+1}) \leq f(\alpha^k)\}| &= 0 \quad \text{or} \\ |\{\gamma^{k-1} \in X \mid \gamma^{k-1} < \alpha^k, f(\gamma^{k-1}) \geq f(\alpha^k)\}| &= 0. \end{aligned} \quad (2.16)$$

Intuitively, a discrete Morse function f allows us to construct a simplicial complex by attaching simplices in a prescribed order. The collection of all simplices with a scalar value smaller than a prescribed threshold $c \in \mathbb{R}$ together with their faces is called *the lower subcomplex* $X(c)$ of X by

$$X(c) = \cup_{f(\alpha) \leq c} \cup_{\beta \leq \alpha} \beta. \quad (2.17)$$

Let us consider two subcomplexes $X(c)$ and $X(d)$ with $X(c) \subset X(d)$ and the property that their difference $X(d) \setminus X(c)$ consists of a pair of simplices (α, β) with $\alpha < \beta$ and β being the only coface of α . We call (α, β) a *free pair* and can obtain the subcomplex $X(c)$ from $X(d)$ by removing the free pair (α, β) . This operation is called *simplicial collapse* [Coh73]. More generally, whenever the removal of a free pair from a simplicial complex \tilde{X} yields again a simplicial complex X , we say \tilde{X} *collapses* to X . If there is a sequence of simplicial collapses and expansions (the inverse of a collapse) transforming \tilde{X} to X , \tilde{X} and X are *simple-homotopy equivalent*. For example, a triangle is simple-homotopy equivalent to a vertex, see Figure 2.9.

Suppose now there are no critical simplices α with $f(\alpha) \in (c, d]$. Then the subcomplex $X(d)$ collapses simplicially to $X(c)$, i.e., both subcomplexes are homotopic equivalent. On the other hand, let α be a critical k -simplex with $f(\alpha) \in (c, d]$. Then we can construct an attaching map $\phi : S^{k-1} \rightarrow X(c)$ such that $X(c) \sqcup_{\phi} e^k$ is homotopic equivalent to $X(d)$. Note that a critical simplex does not belong to any free pair.

The above argumentation sketches the idea of the proof of the main theorem of discrete Morse theory:

Theorem 5 (Main Theorem) *Let f be a discrete Morse function on a simplicial complex X . The complex X is then homotopic equivalent to a cell-complex with exactly one cell of dimension k for each critical simplex of dimension k .*

The main theorem is the discrete analogue to Theorem 4. From a discrete Morse function f given on a simplicial complex X , we can construct a cell complex that consists typically of much fewer cells than X but still carries the same information – its discrete Morse-Smale complex.

From Theorem 5, one can now obtain that the weak and strong Morse inequalities (2.6) and (2.7) also hold for a discrete Morse function. The discrete nature of the cell-complexes and the input function f now enables us to design efficient algorithms to compute the Morse-Smale complex such that the Morse-Smale complex is always consistent with the topology of the underlying cell-complex. In contrast to numerical algorithms, the combinatorial structure herein avoids the need of computational parameters which guarantees the reliability of the computed result.

In the next chapter, we present the algorithmic setting of discrete Morse theory. We use a graph-theoretical formulation to represent the cell-complexes. In this setting, we formulate algorithms allowing to compute the Morse-Smale complex combinatorially as well as different levels of detail of this complex.

Chapter 3

Computational Discrete Morse Theory

In this chapter, we present the algorithmic framework developed in the course of this thesis. This framework allows for a purely combinatorial computation of 2- and 3-dimensional Morse-Smale complexes as introduced in Chapter 2 – no numerical algorithms and computational parameters are needed.

Before we present the algorithmic framework, we provide an overview of the developed algorithms and notations used within these algorithms in Section 3.1. This overview should serve the reader as reference for the subsequent technical details.

In Section 3.2, we introduce the graph theoretical setting in which the cell-complex C is represented. The combinatorial information of C , i.e., the cells and their adjacency, is nicely encoded in an induced cell graph. Given such a graph, we can define a matching on it. Such a matching will allow us to represent the combinatorial gradient flow of a given scalar input. Critical points and separatrices are implicitly defined by it. We present algorithms that enable an efficient computation of these structures by use of their combinatorial properties. These algorithms are the basis for the subsequent algorithms to compute the Morse-Smale complex.

Given the graph theoretical setting, we construct a combinatorial gradient field based on an input function f in Section 3.3. This gradient field encodes the combinatorial flow of f and defines implicitly its Morse-Smale complex. We will use an existing algorithm to compute this gradient field proposed by Robins et al. [RWS11]. The critical points in the resulting gradient field correspond provably 1-1 to the topological changes of the lower level sets. The gradient field is represented as a matching in the induced cell graph, which reduces the memory overhead to a minimum. Only Boolean masks which represent subsets of links and nodes of the graph are needed. This is especially important for the computation of the Morse-Smale complex.

The Morse-Smale complex is only implicitly given by the combinatorial gradient field. To construct an explicit representation, we present in Section 3.4 algorithms that compute the Morse-Smale complex with an optimal complexity of $O(n^2)$ with n denoting the size of the input.

The algorithms of Lewiner [Lew05] and Robins et al. [RWS11] use a classic breadth-first search in the induced cell graph to compute the Morse-Smale complex. While a simple breadth-first search computes the desired complex efficiently in the 2-dimensional case, its computational effort becomes cubic in the 3-dimensional case. In contrast,

	ComputeGradient (Algorithm 5)		
Basis Functionality	BoundaryMatrix (Algorithm 7)	Sequence2D (Algorithm 11)	Sequence3D (Algorithm 14)
GetManifold (Algorithm 1)	GetAllManifolds (Algorithm 8)		InitHeap (Algorithm 15)
AlternatingRestrictedBFS (Algorithm 2)	CountPaths (Algorithm 9)	GetMinWeight2D (Algorithm 12)	GetMinWeight3D (Algorithm 16)
AlternatingEdges (Algorithm 3)	GetManifoldNodes (Algorithm 10)	GetPath2D (Algorithm 13)	GetPath3D (Algorithm 17)
GetIntersection (Algorithm 4)			IsUnique (Algorithm 18)

Table 3.1: Algorithmic overview of the computational framework.

our algorithm restricts the search space to the essential links of the graph. Due to this restriction, the computational effort for the Morse-Smale complex becomes quadratic, which is provably the optimal computational complexity in three dimensions.

After presenting the algorithms to compute the Morse-Smale complex, we introduce a strategy to create a hierarchy representing different levels of detail of the complex for the 2- as well as the 3-dimensional case in Section 3.5. Since sampling artifacts and small fluctuations in the data may create spurious structures in the Morse-Smale complex, we can use this strategy to differentiate spurious structures from those corresponding to the large-scale behavior of the function f . We greatly benefit from the implicit representation herein. An explicit representation of the Morse-Smale complex would require special data structures. While the design of efficient data structures is already a non-trivial task, additional heuristics need also to be introduced to manage the quadratic memory consumption in the hierarchy computation as shown by Gyulassy [Gyu08]. Since these heuristics introduce computational parameters, the result heavily depends on it; and the parameters need to be adjusted for each data set. The algorithms developed in the course of this work need no computational parameters, and therefore, allow for an user-independent multi-level representation of the Morse-Smale complex.

To the best of our knowledge, our algorithms combine for the first time a linear memory consumption and a running time which behaves well in practice. Due to these properties, large data sets can be analyzed on commodity hardware. The presented algorithms are peer-reviewed and published in [RGH⁺10], [GRP⁺11], [GRWH11], and [GRWH12].

3.1 Preliminaries

Before we start with the description of the algorithmic framework, we provide an overview of the developed algorithms and notation used within these algorithms. The basic idea of the presented framework is to keep it as modular as possible to allow for a simple exchange of algorithms. Some of the algorithms are therefore relatively small. The algorithms are designed to enable a straight-forward implementation using the STL [MS95]. Only set operations are needed. However, performance improvements can be achieved by using optimized containers that allow optimal access and enable an efficient parallel computation provided by the BOOST library [Kar05]. Although

Notation	Relation	Discription
G	$G = (N, E)$	cell-graph
N		nodes of G
E		links of G
E_k	$E_k \subset E$	links of index k
N_T	$N_T = N(T) \subset N$	nodes covered by links $T \subset E$
E_M	$E_M = E(M) \subset E$	links covered by nodes $M \subset N$
V	$V \subset E$	combinatorial gradient field
S	$S \subset E$	integrated manifold
R	$R \subset E$	restriction of integration
I	$I \subset E$	intersection of manifolds
C^k	$C^k \subset N$	critical nodes of index k in V
c^k	$c^k \in C^k$	critical nodes of index k
C_T	$C_T \subset N$	critical nodes covered by $T \subset E$
ℓ	$\ell \in \{0, 1, 2\}$	layer of integration
j	$j \in \{0, 1\}$	direction of integration
∂_k	$\partial_k : C^k \rightarrow C^{k-1}$	k -th boundary matrix
Q		queue
h		heap (priority queue)

Table 3.2: Overview of the notations used in the algorithms.

most of the notations are formally introduced in the subsequent sections, the following overview should serve as reference and help the reader in the technical details of the algorithms.

In the course of this chapter, three main algorithms are presented:

1. BoundaryMatrix (Algorithm 7),
2. Sequence2D (Algorithm 11),
3. Sequence3D (Algorithm 14).

Algorithm 7 computes the boundary operator ∂_k that induces the Morse-Smale complex C^M (2.13). The output is a sparse matrix representing the adjacency of the critical points. Section 3.4 describes in detail how this computation can be done efficiently. A multi-level representation of the Morse-Smale complex is computed by Algorithms 11 and 14 for the 2- and 3-dimensional case, respectively. The technical aspects are described in the Sections 3.5.1 and 3.5.3.

There are some elementary procedures that provide the functionality that manifolds can be integrated and intersected. This functionality is essential and used by almost all algorithms. The core of the framework consists of the functions

AlternatingRestrictedBFS (Algorithm 2), and
AlternatingEdges (Algorithm 3).

These two functions provide the technical part of the integration – the restricted breadth-first search – of the ascending and descending manifolds and/or subsets of them. The overall integration and intersection of these manifolds is done by the functions

GetManifold (Algorithm 1), and
GetIntersection (Algorithm 4).

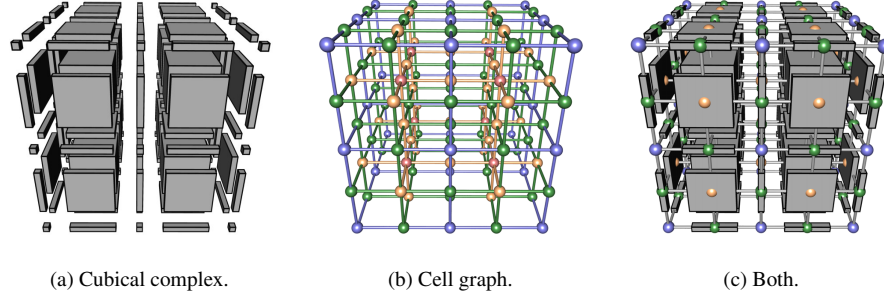


Figure 3.1: Illustration of a cell complex and its derived cell graph. a) shows the cells of a $2 \times 2 \times 2$ uniform grid in an exploding view. A single voxel is represented by eight 0-cells, twelve 1-cells, six 2-cells, and one 3-dimensional cell. These cells and their boundary relation define the cell complex. b) shows the derived cell graph. The nodes representing the 0-, 1-, 2-, and 3-cells are shown as blue, green, yellow and red spheres respectively. The adjacency of the nodes is given by the boundary relation of the cells. The edges are colored by the lower dimensional incident node. c) shows the cell complex and the cell graph to illustrate the neighborhood relation of the cells.

Besides these elementary procedures, the main algorithms 7, 11, and 14 use specific sub-functions to complete their task. For example, the number of paths between two critical points needs to be counted to compute the boundary matrix ∂_k with coefficients in \mathbb{Z}_2 . This functionality is provided by Algorithm 9 and called by the main algorithm BoundaryMatrix. An overview of all algorithms used within the framework is given in Table 3.1.

The input of the main algorithms is always a combinatorial gradient field. We use an existing algorithm proposed by Robins et al. [RWS11] to compute it. For completeness, we provide a description of this algorithm in our graph-theoretical notation in Section 3.3 and pseudo-code in Algorithm 5.

The subsequent algorithms compute subsets of nodes and links of a graph. Since most of them appear frequently in many algorithms, Table 3.2 lists the most important notations as reference.

3.2 The Graph Theoretical Setting

In the following, we assume that the input domain $\Omega \subset \mathbb{R}^3$ is given as a simplicial or cubical complex. We only consider 2- and 3-dimensional complexes. Following the notation from Section 2.1, we denote the induced chain complex by C .

3.2.1 The Cell Graph

The combinatorial setting of Section 2.4 can be nicely represented in a graph theoretical formulation: the *cell graph* $G = (N, E)$ encodes the essential combinatorial information of the chain complex C . The *nodes* N represent the basis elements of the free abelian groups C_k , i.e., the k -cells of X . We use the barycenter of a cell as the geometric embedding of the representing node. Each node u^k is labeled by the dimension k of the cell it represents. For simplicity, the notion of u^k is used for representing the nodes as

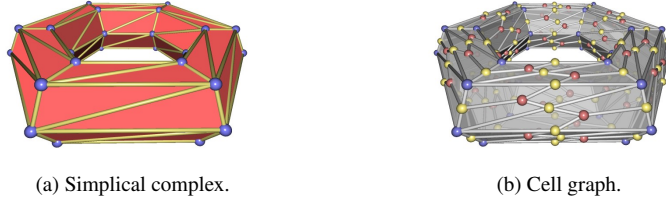


Figure 3.2: Illustration of a simplicial complex and its derived cell graph. (a) shows the cells of a 2-dimensional simplicial complex. (b) shows the derived cell graph. The nodes representing the 0-, 1-, and 2-cells are shown as blue, yellow and red spheres respectively. The adjacency of the nodes is given by the boundary relation of the cells.

well as the corresponding cells. The boundary operator ∂ (2.1) defines the adjacency information of the nodes. We represent the adjacency by the *links* E of G . If a cell u^k is in the boundary of a cell w^{k+1} , then $e^k = \{u^k, w^{k+1}\} \in E$. We label each link with the dimension of the lower dimensional node. The link e^k is said to be of index k . An illustration of a cell graph of a cubical complex is shown in Figure 3.1. An example of a cell graph induced by a simplicial complex is given in Figure 3.2.

We want to stress that the cell graph G is closely related to a Hasse diagram, see Figure 3.3. Each layer ℓ corresponds to the nodes of index k . However, the geometric embedding of the cells is naturally given in G . From the data analysis point of view, such kind of representation is preferable.

The cell graph of a simplicial complex is explicitly represented. The node- and link indices of the graph are stored in arrays allowing direct access. The adjacency of the nodes is given by the sparse boundary matrix ∂ as defined in Section 2.1. The memory consumption of the cell graph representation is thereby proportional to the number of cells in the simplicial complex.

The cell graph of a cubical complex, on the other hand, can be implicitly represented which avoids prohibitively large amounts of memory. We follow here the work of Kovalevsky [Kov01]. The graph G is represented as a regular grid and the node- and link indices of G are calculated on-the-fly based on the resolution of the cubical complex. A node's index is computed as the sum of the parities of its integer coordinates. Links connect two nodes along one of the coordinate axes. We associate each link with the end node of lower index along this coordinate axis and lay out the nodes

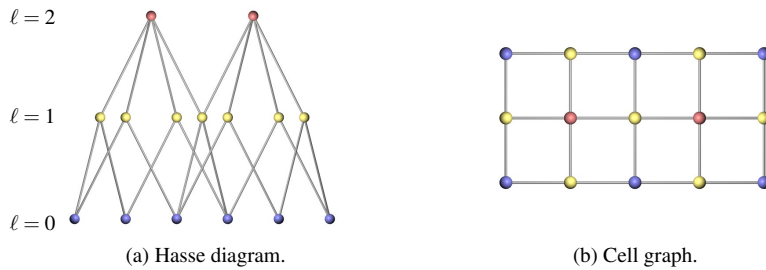


Figure 3.3: Two graph representations of a 2-dimensional cubical complex. The nodes are colored by their index: blue ($k = 0$), yellow ($k = 1$), red ($k = 2$).

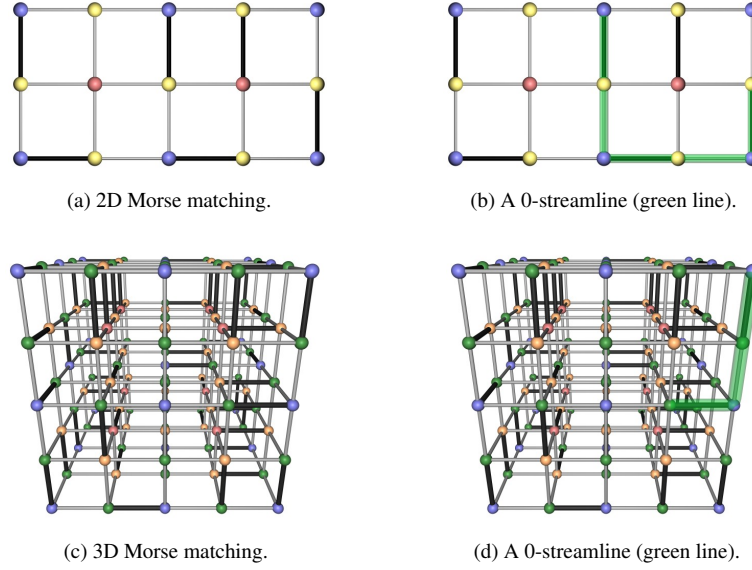


Figure 3.4: Morse matching: The edges of the Morse matching are shown as black solid lines. (a) shows a Morse matching of a 2D cell graph. The nodes of index 0, 1, and 2 are shown as blue, yellow, and red spheres. (c) shows a Morse matching of a 3D cell graph. The nodes of index 0, 1, 2, and 3 are shown as blue, green, yellow, and red spheres. (b) and (d) show a 0-streamline in the corresponding Morse matchings.

and links separately in raster scan order. For nodes, this is the usual array layout. For the links, we consider first a 2-dimensional example as shown in Figure 3.3b. Due to the regular structure of the grid, a pattern is formed by the links that repeats only every second row: links in first row point along x and links in second row point along y . For the 3-dimensional case as shown in Figure 3.1b, the situation is the same. The 2-dimensional pattern repeats every second slice: links in first slice point along x and y and links in second slice point along z .

3.2.2 Morse Matchings

The translation of an arbitrary input function to the axioms of a discrete Morse function (2.15) can be done in several ways. One way to do it, is the lower-star filtration defined in Section 3.3. However, we are primarily not interested in the discrete Morse function but in its combinatorial gradient flow. Not only are the critical points and separatrices uniquely defined by this flow, it also allows for an efficient integration of combinatorial streamlines. In our graph theoretical setting, the combinatorial flow can best be represented by a matching. Chari [Cha00] showed that a specific class of matchings, which is defined in the next paragraph, is in a 1-1 correspondence with the set of discrete Morse functions.

A *matching* $V \subset E$ is a collection of links in G such that none of these links are adjacent. The matching V gives rise to a combinatorial flow. Assuming the cell graph G is initially oriented from top to bottom, i.e., the higher dimensional nodes point to the lower dimensional ones, the matching V now reverses the orientation of the links $e \in V$. This allows a traversal of the cell graph G . Given a link e , a *combinatorial*

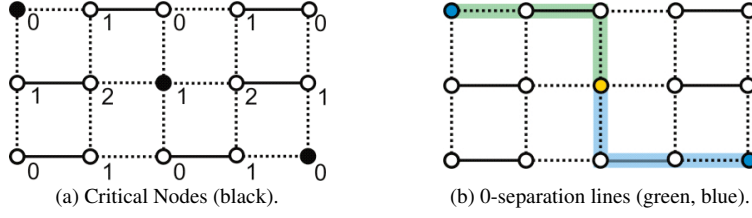


Figure 3.5: Illustration of the critical nodes and separation lines of a 2D Morse matching. The links of the Morse matching are depicted as solid black lines. The nodes of the cell graph in (a) are labeled by their index. The saddle and the minima in (b) are shown as yellow and blue spheres, respectively.

k -streamline is an alternating sequence of links of index k in V and its complement $E \setminus V$ beginning with e . If there are no closed k -streamline in G , i.e., the intersection of the start- and end-link is empty, V is an acyclic matching. We call such an acyclic matching V a *Morse matching* [Cha00]. Figure 3.4 illustrates a 2- and 3-dimensional Morse matching and depicts a 0-streamline within.

The main task in computational discrete Morse theory is now to construct a Morse matching V such that the following combinatorial definitions of critical points and separatrices correspond to an input function f . We will show in the subsequent sections how such a matching can be efficiently constructed. To emphasize that the input is a scalar function, we call V also a *combinatorial gradient field*.

3.2.3 Critical Points and Combinatorial Separation Lines

In the combinatorial setting, the *critical nodes* are the unmatched nodes of V , i.e., the nodes of the graph with no incident matched links. For a cell graph induced by a 2-dimensional cell complex, we call a critical node u^k of index k a minimum ($k = 0$), a saddle ($k = 1$), or a maximum ($k = 2$). In case of a 3-dimensional cell graph, the critical nodes are called: minimum ($k = 0$), 1-saddle ($k = 1$), 2-saddle ($k = 2$), or maximum ($k = 3$). Given an initial orientation as described in Section 3.2.2, the critical nodes have a similar interpretation as their smooth analogue: the 0-streamlines end in minima, the 2-streamlines emanate from maxima, and at saddles we observe a mixed behavior of the combinatorial flow.

A k -streamline connecting two critical nodes u^k and w^{k+1} is called a *combinatorial k -separation line*. For a 2-dimensional cell graph, there are two types of lines: the 0-separation lines connect the saddles with the minima, and the 1-separation lines connect the saddles to their maxima. Figure 3.5 shows an illustration of two 0-separation lines. For a 3-dimensional cell graph, the 0- and 2-separation lines have a similar role: the 1-saddles are connected to the minima, and the 2-saddles to the maxima, respectively. However, there are also 1-separation lines connecting the two types of saddles.

For the special case of the 3-dimensional cell graphs, we want to emphasize the 2-dimensional ascending and descending combinatorial manifolds emanating from the two types of saddles. We call these objects *combinatorial separation surface* given by all 1-streamlines emanating from a 1- or 2-saddle. We will need these surfaces later on to compute efficiently the 1-separation lines. Note that combinatorial separation lines and surfaces are given as discrete sets of links in G in contrast to their continuous counterparts (2.9) and (2.12).

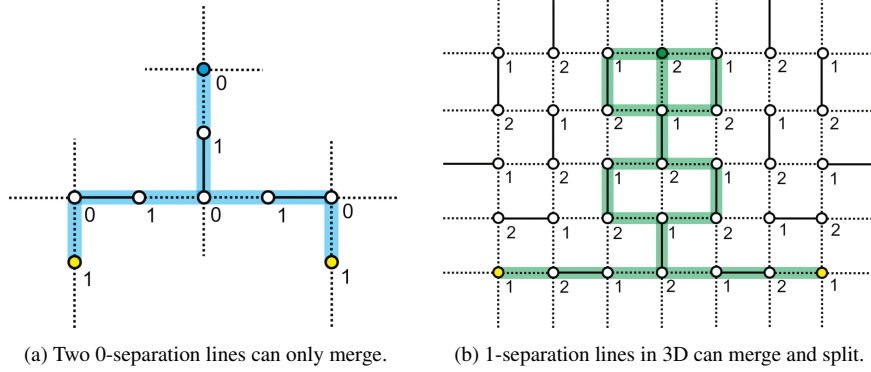


Figure 3.6: Properties of separation lines. The edges of the Morse matching are shown as solid black lines. Only the essential parts of the cell graph are shown. The nodes of the cell graph are labeled by their index. The saddles in (a) are shown as yellow spheres while the minimum is depicted as blue sphere. The repelling saddle in (b) is shown as green sphere while the two attracting saddles are shown as yellow sphere.

3.2.4 Properties of Combinatorial Separation Lines

Using (2.8) and (2.9), the separation lines are given as the intersection of the corresponding combinatorial ascending and descending manifolds. We only need to replace the continuous negative flow ϕ with its combinatorial analogue. The combinatorial flow is restricted to the links of the cell graph and is represented by the Morse matching. In contrast to the smooth setting of Section 2.2, the separation lines can merge and split due to their combinatorial nature.

To compute the separation lines, an integration of the ascending and descending manifolds can be computationally expensive. However, we can make use of the specific structure of these manifolds to compute the separation lines connecting saddles to the extrema. In this special case, the descending/ascending manifolds degenerate to lines and are uniquely defined by a breadth-first search starting at the saddles. No intersection is necessary. This stems from the fact that those lines can *either merge or split*.

Consider for example a 0-separation line starting at a saddle as shown in Figure 3.6a. The 0-separation line covers 0- and 1-nodes of the cell graph. The 1-nodes represent the edges of the cell complex. Since a matching is defined such that no two matched links are adjacent, two 0-separation lines can only merge at a 0-node. A splitting of them would require that there exists at least two outgoing nodes. However, each edge is always bounded by exactly two vertices. A 0-separation line can only enter at one vertex and leave on the other vertex.

A similar argumentation holds for the separation lines connecting the saddles with the maxima. For a 2-dimensional cell graph, a 1-node is always connected to at most two 2-nodes. A 2-node in a 3-dimensional cell graph, on the other hand, is always connected to at most two 3-nodes, i.e., a square or triangle in a cubical/simplicial complex is only in the boundary of at most two cubes or tetrahedrons, respectively. However, these separation lines have a special characteristic compared to the 0-separation lines. They can end in the boundary of the complex such that the end-node of the line is not critical. In this case, the corresponding bounding cell contains only a single coface.

We call such lines *virtual combinatorial separation lines*.

The critical nodes are easily collected by iterating over all nodes of the cell graph and checking whether the incident links are matched. The separation lines that connect the saddles with the extrema can be computed using Algorithm 1, 2, and 3.

Algorithm 1 GetManifold(G, V, c^p, ℓ)

Input: $G = (N, E), V \subset E, c^p \in N, \ell \in \{0, 1, 2\}$

Output: $S \subset E$

- 1: $E_\ell \leftarrow \{e^k \in E : k = \ell\}$ \triangleleft set of links of index ℓ
 - 2: $S \leftarrow \text{AlternatingRestrictedBFS}(G, V, E_\ell, c^p)$ \triangleleft integrate with restriction E_ℓ
-

The input of Algorithm 1 is the cell graph G , a critical node c^p of index p and the index ℓ of the links that should be integrated. We call the index ℓ also *layer of integration*. The output is a set of links S representing the corresponding ascending or descending manifolds emanating at a critical node c^p (line 2). The pseudo-code of Algorithm 1 as well as of the following algorithms is given in its most general form, i.e., not restricted to a specific index or layer. In the special case discussed above, the integrated manifolds of Algorithm 1 coincide with the separation lines if the input critical node c^p is a saddle.

Algorithm 2 AlternatingRestrictedBFS(G, V, R, c^p)

Input: $G = (N, E), V \subset E, R \subset E, c^p \in N$

Output: $T \subset R \subset E$

- 1: $T \leftarrow \emptyset, Q \leftarrow \emptyset$ \triangleleft set of integrated links
 - 2: $Q.\text{push}(\{c^p, \text{false}\})$ \triangleleft initialize
 - 3: **while** $Q \neq \emptyset$ **do** \triangleleft breadth-first search
 - 4: $\{u^p, \text{flag}\} \leftarrow Q.\text{pop}()$ \triangleleft get the next node and the flag of its links
 - 5: $W \leftarrow \text{AlternatingEdges}(G, V, u^p, \text{flag})$ \triangleleft get the correctly flagged links
 - 6: $W \leftarrow (W \cap R) \setminus T$ \triangleleft apply restriction R and remove visited links T
 - 7: **for all** $\{u^p, w^k\} \in W$ **do**
 - 8: $T \leftarrow T \cup \{u^p, w^k\}$ \triangleleft add links to the output
 - 9: $Q.\text{push}(\{w^k, \neg \text{flag}\})$ \triangleleft push next node with negated link flag
-

The integration is done by a breadth-first search restricted to the layer ℓ shown in Algorithm 2. Since all links incident to a critical node are unmatched, the integration starts with an unmatched link (line 2). The combinatorial ℓ -manifolds consist of ℓ -streamlines which alternate between the Morse matching V and its complement. This needs to be respected by the integration (lines 5 and 9, and Algorithm 3). However, only those links are of interest that lie in the current layer and are not yet visited (line

Algorithm 3 AlternatingEdges(G, V, u^p, flag)

Input: $G = (N, E), V \subset E, u^p \in N, \text{flag} \in \{\text{false}, \text{true}\}$

Output: $W \subset E$

- 1: **if** $\text{flag} = \text{true}$ **then**
 - 2: $W \leftarrow \{\{u^p, w^k\} \in E : \{u^p, w^k\} \in V\}$ \triangleleft links belong to V
 - 3: **else**
 - 4: $W \leftarrow \{\{u^p, w^k\} \in E : \{u^p, w^k\} \in E \setminus V\}$ \triangleleft links belong to $E \setminus V$
-

6). If there is such a link (line 7, 8 and 9), the integration continues (line 3), otherwise it stops.

We now consider the special case of 1-separation lines in a 3-dimensional cell graph. Those lines connect saddles of different index. In contrast to the 0- and 2-separation lines, they can merge *and* split. Figure 3.6b illustrates a typical situation. This stems from the specific structure of a 3-dimensional cell complex: an edge in a cubical or simplicial complex has more than two cofaces. Consequently, a 1-node is connected to more than two 2-nodes. A 1-streamline can therefore enter a 1-node from one direction, and leave it in several other directions. This property prohibits a direct walk starting at a saddle as we have done it for the 0- and 2-separation lines.

To compute 1-separation lines in a 3-dimensional cell graph, we can make use of the formal definition of separation lines: the intersection of the ascending and descending manifolds (2.9). We first integrate the separation surface emanating from a saddle using Algorithm 1. The intersection is then computed by Algorithm 4 using the separation surface as restriction in the breadth-first search (Algorithm 2).

The computation of the 1-separation lines is symmetric: either one computes the descending manifolds of all 2-saddles and intersects them with the ascending manifolds of the 1-saddles, or vice versa. The flag j in Algorithm 4 defines this direction of intersection. For simplicity, we only discuss the case for $j = 0$ in the following, i.e., we first compute the descending manifold of a 2-saddle c^2 . The layer of integration is $\ell = 1$.

Algorithm 4 GetIntersection(G, V, S, ℓ, j)

Input: $G = (N, E), V \subset E, S \subset E, \ell \in \{0, 1, 2\}, j \in \{0, 1\}$
Output: $I \subset E$ \triangleleft intersection of two manifolds
1: $C_S \leftarrow \{u^{\ell+j} \in N : \exists \{u^{\ell+j}, w^k\} \in S\} \cap C_{\ell+j}$ \triangleleft critical nodes in the boundary of S
2: $I \leftarrow \emptyset$ \triangleleft initialize intersection
3: **for all** $c^p \in C_S$ **do** \triangleleft for each boundary critical node
4: $S \leftarrow S \setminus I$ \triangleleft remove already visited links from the restriction S
5: $I \leftarrow I \cup \text{AlternatingRestrictedBFS}(G, V, S, c^p)$ \triangleleft apply back-integration

The set of links $S \subset E$ describing the separation surface that emanates from a 2-saddle c^2 (line 2) is computed using Algorithm 1. Algorithm 4 computes then the intersection I of the manifolds of the adjacent 1-saddles with respect to S . The integration starts with the complementary 1-saddles. We therefore collect all 1-saddles $C_S \subset C^1$ that are covered by S (line 1). These critical nodes are connected to c^2 . Finally, we compute the set of links $I \subset S$ of all combinatorial streamlines connecting c^2 with C_S (line 3). Naïvely, one would compute the entire ascending manifold of all 1-saddles C_S and intersect the corresponding sets of links. However, this is not necessary. We only need to integrate those parts that are also part of the descending manifold S of c^2 . The intersection is always a subset of the links of the ascending as well as of the descending manifolds. This is done by a restricted breadth-first search (line 5). Note that each link $e \in I$ is only visited once. The restriction is iteratively updated (line 4).

We want to note that Algorithm 4 can also be used to compute the separation lines connecting the saddles with the extrema. However, the computation of the intersection needs more computational effort than a direct walk starting at a saddle.

Having defined the combinatorial analogue of critical points and separation lines, we can now investigate how a combinatorial gradient field $V \subset E$ can be constructed based on an input function f .

3.3 Computation of Combinatorial Gradient Fields

The input function f usually assigns only scalar values to the 0-cells of a simplicial/cubical complex X . Since each cell needs to be assigned with a number, we first extend f to all cells of X by a so-called *lower-star filtration*: each cell is assigned the maximum function value of the vertices it contains. This yields a function $\tilde{f} : N \rightarrow \mathbb{R}$ defined on nodes N of the cell graph $G = (N, E)$.

As already mentioned, the main task in computational discrete Morse theory is to construct a combinatorial gradient that corresponds to the input data f . Many such algorithms have been proposed [BLW12, GBPH11, KKM05, Lew05]. In this work, we make use of the algorithm *ProcessLowerStars* proposed by Robins et al. [RWS11]. The critical nodes of their combinatorial gradient provably correspond 1-1 to the topological changes of the lower-level sets (Section 2.2) of the input data in up to three dimensions. Other algorithms [VS91, Gyu08, SN12] are not able to make such guarantees about the number of critical nodes in their gradient field. It may happen that there is a large number of false positives due to the algorithmic design, which would complicate or mislead a further analysis of the data. Also, the algorithm of Robins et al. is very efficient, since it has linear running time and allows for an efficient parallel implementation. The pseudo-code of this algorithm in our graph theoretical notation is

Algorithm 5 CombinatorialGradient(G, \tilde{f})

Input: $G = (N, E)$, $\tilde{f} : N \rightarrow \mathbb{R}$

Output: $V \subset E$

```

1:  $V \leftarrow \emptyset$                                  $\triangleleft$  initialize
2: for all  $v^0 \in N$  do                         $\triangleleft$  for each 0-node
3:    $s \leftarrow v^0$                              $\triangleleft$  create its lower star
4:    $W \leftarrow \{w \in N : \tilde{f}(w) \leq \tilde{f}(v^0)\}$ 
5:   for  $p \leftarrow 0, \dots, d-1$  do
6:      $s \leftarrow s \cup \{w^{p+1} \in W : \exists \{u^p, w^{p+1}\} \in E, u^p \in S\}$ 
7:      $K \leftarrow E(s)$                          $\triangleleft$  get the links connecting the lower star nodes
8:      $C \leftarrow \emptyset$                          $\triangleleft$  initialize list of flagged nodes
9:      $abort \leftarrow false$ 
10:    while  $abort = false$  do                     $\triangleleft$  start homotopic expansion
11:       $abort \leftarrow true$ 
12:      for  $p \leftarrow 0, \dots, d-1$  do             $\triangleleft$  for each dimension
13:         $T \leftarrow \{\{u^p, w^{p+1}\} \in K : u^p, w^{p+1} \notin C \cup N(V)\}$   $\triangleleft$  nodes are not covered
14:         $L \leftarrow T \setminus \{\{u^p, w^{p+1}\} \in T : \exists \{z^p, w^{p+1}\} \in T, u^p \neq z^p\}$   $\triangleleft$  and unique
15:        if  $L \neq \emptyset$  then                     $\triangleleft$  there are valid expansions
16:           $V \leftarrow V \cup \text{ChooseLink}(L)$        $\triangleleft$  apply homotopic expansion
17:           $abort \leftarrow false$ 
18:          goto line 10                           $\triangleleft$  continue homotopic expansion
19:        else                                     $\triangleleft$  there is no valid expansions
20:          for  $k \leftarrow 0, \dots, d$  do           $\triangleleft$  for each dimension
21:             $\{u_0^k, u_1^k, \dots, u_m^k\} \leftarrow \{u^k \in S : u^k \notin C \cup N(V)\}$   $\triangleleft$  unflagged nodes
22:            if  $\{u^k \in s : u^k \notin C \cup N(V)\} \neq \emptyset$  then
23:               $C \leftarrow C \cup u_0^k$              $\triangleleft$  flag an arbitrary unflagged node
24:               $abort \leftarrow false$ 
25:              goto line 10                       $\triangleleft$  continue homotopic expansion

```

given in Algorithm 5.

Algorithm 5 decomposes the cell graph G into the lower stars [Ban70] defined by \tilde{f} (line 3, 4, 5 and 6). Note that this decomposition is disjoint, which allows for good parallel scalability. Each lower star is now grown from its vertex using *simple homotopic expansions* – the inverse of a homotopic collapse described in Section 2.4. Such expansions are represented by the links of the cell graph (line 7).

The combinatorial gradient V is now constructed iteratively (line 10): each time we expand a lower star using a link $e \in E$ (lines 13, 14 and 15), we append e to V (line 16). A link $e = \{u^p, w^{p+1}\} \in E$ is admissible for simple homotopic expansion if the following conditions hold:

1. u^p and w^{p+1} are not covered by a link in the current V (line 13),
2. u^p and w^{p+1} have not been flagged previously (line 13),
3. there is no other link $\{z^p, w^{p+1}\} \in E$ that fulfills 1. and 2. (line 14).

If the set of admissible links L is empty (line 19), we flag an arbitrary node in the lower star (line 23) that is not covered by a link in the current V (line 22). If no such node can be found, the expansion stops. If L is not empty (line 15), an admissible link is chosen based on an order defined by \tilde{f} and appended to V (line 16).

As shown by Robins et al. [RWS11], the way we choose a link from L (line 16 of Algorithm 5) does not affect the overall number nor the type of critical nodes in the resulting combinatorial gradient. However, the combinatorial streamlines are affected by this choice. Since the combinatorial gradient is supposed to correspond to the (continuous) gradient of the input, a natural choice is the link that represents locally the steepest descent. However, other choices are also possible. In Chapter 6, we will discuss a probabilistic alternative to the steepest descent.

Performance

In the following, we present some examples to illustrate the running time of Algorithm 5 for real-world data. The experiments were done on an Intel Core i7-2720QM CPU with 16GB RAM. To compute the combinatorial gradient field, we implemented an OpenMP version of Algorithm 5. Table 3.3 shows the running time for different 3-dimensional data sets. The neghip, hydrogen and aneurism are provided by *The Volume Library* [Röt] while the beetle data set is provided by Gröller et al. [GGK]. We give the running time for the computation of the combinatorial gradient field using Algorithm 5 with 4 physical (8 logical) cores and 1 core, respectively, and the corresponding speed up factor.

Data Set (Size)	Neghip (64 ³)	Hydrogen (128 ³)	Aneurism (256 ³)	Beetle (416 ² × 247)	Benzene (401 ³)	Synthetic (1024 ³)
Single Core	11s	1m 22s	10m 40s	27m 10s	41m 15s	–
4 × 4 Cores	2s	18s	2m 28s	6m 36s	10m 01s	165m 23s
Speed Up	5.5	4.5	4.3	4.1	4.1	–

Table 3.3: Running times of Algorithm 5 for six data sets of varying dimensions. The first row shows the running time to compute the combinatorial gradient field using single threaded computation while the second row shows the times using the 4 physical and 8 logical cores. The resulting speed up factor is shown in the third row.

The speed up factor using a parallel implementation of Algorithm 5 is nearly optimal. Its running times increase linearly with the size of the data set independently of the topological complexity of the data.

3.4 Computation of Morse-Smale Complexes

The combinatorial gradient field V only implicitly encodes the Morse-Smale complex. In the following, we describe how an explicit representation of the Morse-Smale complex (2.13) can be computed from V with an optimal complexity of $O(n^2)$ with n denoting the number of vertices of the cell complex C . We only describe the algorithms for the 3-dimensional case since this is the more difficult part. However, they can also be directly applied to 2-dimensional cell graphs.

The crucial part is to count optimally the number of separation lines between two critical nodes thereby avoiding multiple traversals of the links E . Since we use coefficients in \mathbb{Z}_2 , two critical nodes are adjacent in the sense of \mathbb{Z}_2 if there is an odd number of separation lines connecting them.

The chain groups C^ℓ of the Morse-Smale complex (2.13) can be easily extracted from N by collecting the nodes not covered by V . However, the efficient computation of the boundary maps $\partial_\ell : C^\ell \rightarrow C^{\ell-1}$ is challenging.

Before we present our optimal algorithm to compute ∂_ℓ with a complexity of $O(n^2)$, we describe a straight-forward realization proposed by Robins et al. [RWS11] with a complexity of $O(n^3)$. The algorithms presented in the following make use of the basic algorithms described in Section 3.2.3. Following the notation of Section 3.2, the cell graph is denoted by $G = (N, E)$ and its combinatorial gradient field by $V \subset E$.

3.4.1 The Algorithm by Robins et al.

Algorithm 6 shows a simple approach introduced by Robins et al. [RWS11] to compute $\partial_{\ell+1}$ with a worst case complexity of $O(n^3)$, where n denotes the number of vertices of C . Its input consists of the cell graph $G = (N, E)$, a discrete gradient field $V \subset E$, a flag j denoting the (ascending/descending) direction of integration, and the layer of integration $\ell \in \{0, 1, 2\}$ yielding the boundary map $\partial_{\ell+1}$. If $j = 0$, the algorithm computes $\partial_{\ell+1}$ by finding the boundaries of the elements contained in $C^{\ell+1}$. If $j = 1$, the algorithm computes $\partial_{\ell+1}$ by finding the co -boundaries of the elements contained in C^ℓ . Note that both cases result in the same $\partial_{\ell+1}$, the choice of j only affects the running time. In Section 3.4.4, we discuss which choice of j is appropriate, in practice.

The main idea of Algorithm 6 is to start a breadth-first search in each critical node c^p . The integration of the combinatorial manifold stops at its boundary. The critical nodes in the boundary are adjacent to c^p if there is an odd number of paths connecting each of the boundary nodes with c^p . Due to the combinatorial nature, the manifolds of multiple critical nodes c^p can merge. Iterating over the critical nodes (line 3) results therefore in multiple traversals of the links and causes the cubic complexity.

The breadth-first search (line 5) is constrained by the definition of a combinatorial streamline – the links of a traced path must always alternate between V and $E \setminus V$ (line 7). Note that the additions in the lines 13 and 15 are modulo 2.

Algorithm 6 Robins et al. [RWS11] (G, V, j, ℓ)

Input: $G = (N, E), V \subset E, j \in \{0, 1\}, \ell \in \{0, 1, 2\}$

Output: Binary matrix $\partial_{\ell+1}$ \triangleleft boundary matrix

```

1:  $Q \leftarrow \emptyset$   $\triangleleft$  initialize
2:  $E_\ell \leftarrow \{e^k \in E : k = \ell\}$   $\triangleleft$  links of index  $\ell$ 
3: for all  $c^p \in C^{\ell+1-j}$  do  $\triangleleft$  for each critical node of index  $\ell+1-j$ 
4:    $Q.push(\{c^p, false\})$   $\triangleleft$  the links of the start node are unmatched
5:   while  $Q \neq \emptyset$  do  $\triangleleft$  breadth-first search
6:      $\{u^p, flag\} \leftarrow Q.pop()$   $\triangleleft$  get the next node and the flag of its links
7:      $W \leftarrow AlternatingEdges(G, V, u^p, flag)$   $\triangleleft$  get the correctly flagged links
8:      $W \leftarrow W \cap E_\ell$   $\triangleleft$  consider only links of index  $\ell$ 
9:     for all  $\{u^p, w^k\} \in W$  do
10:        $Q.push(\{w^k, \neg flag\})$   $\triangleleft$  push next node with negated link flag
11:       if  $w^k \in C_V$  then  $\triangleleft$  if the node is critical
12:         if  $k < p$  then  $\triangleleft$  the node is a boundary node
13:            $\partial_{\ell+1}(c^p, w^k) \leftarrow \partial_{\ell+1}(c^p, w^k) + 1$   $\triangleleft$  addition is modulo 2
14:         else  $\triangleleft$  the node is a coboundary node
15:            $\partial_{\ell+1}(w^k, c^p) \leftarrow \partial_{\ell+1}(w^k, c^p) + 1$   $\triangleleft$  addition is modulo 2

```

3.4.2 An Optimal Algorithm

We now present our improved Algorithm 7 to compute $\partial_{\ell+1}$ with a worst case complexity of $O(n^2)$. The main idea is the following: We first collect all critical (unmatched) nodes in V . For each of these nodes, we then integrate the corresponding manifolds to collect the critical nodes in the respective (co-)boundaries but avoid multiple traversals of the manifolds (line 1). Since the connections between critical nodes are defined as the intersection of their manifolds, we apply a backintegration for each of the (co)boundary nodes restricted to the already integrated manifold (line 2). This results in a set of links describing all connections between critical nodes.

The challenging task is now to check whether a pair of critical nodes is connected by an odd number of separatrices (line 4). If this is the case, these nodes are connected in the sense of \mathbb{Z}_2 and are inserted in the boundary matrix (line 7 and 9). To count the number of connections, we compute the multiplicity of paths from one critical node to another critical node but restricted to the intersection of the corresponding manifolds.

Algorithm 7 BoundaryMatrix(G, V, j, ℓ)

Input: $G = (N, E), V \subset E, j \in \{0, 1\}, \ell \in \{0, 1, 2\}$

Output: Binary matrix $\partial_{\ell+1}$ \triangleleft boundary matrix

```

1:  $S \leftarrow GetAllManifolds(G, V, \ell, j)$   $\triangleleft$  integrate all manifolds
2:  $I \leftarrow GetIntersection(G, V, S, \ell, j)$   $\triangleleft$  compute intersection w.r.t. the boundary nodes
3: for all  $c^p \in C^{\ell+1-j}$  do  $\triangleleft$  for all critical nodes of index  $\ell+1-j$ 
4:    $C_c \leftarrow CountPaths(G, V, I, c^p, \ell)$   $\triangleleft$  compute boundary nodes  $C_c$  w.r.t.  $\mathbb{Z}_2$ 
5:   for all  $w^k \in C_c$  do
6:     if  $k < p$  then  $\triangleleft$  the node is a boundary node
7:        $\partial_{\ell+1}(c^p, w^k) \leftarrow \partial_{\ell+1}(c^p, w^k) + 1$   $\triangleleft$  add node to  $\partial_{\ell+1}$ 
8:     else  $\triangleleft$  the node is a coboundary node
9:        $\partial_{\ell+1}(w^k, c^p) \leftarrow \partial_{\ell+1}(w^k, c^p) + 1$   $\triangleleft$  add node to  $\partial_{\ell+1}$ 

```

The input of Algorithm 7 is similar to the input of Algorithm 6 and consists of the cell graph $G = (N, E)$, a discrete gradient field $V \subset E$, a flag j , and the index ℓ of the resulting boundary map $\partial_{\ell+1}$. For notational simplicity in the following explanation, we only describe the algorithms in detail for $j = 0$ – we consider the boundary of $c^{\ell+1} \in C^{\ell+1}$. However, all algorithms are designed to work also for $j = 1$. The choice of j does not affect the overall computational complexity – it only affects the practical running time of the algorithm, see Section 3.4.4.

The links $S \subset E$ covered by the combinatorial ℓ -streamlines emanating from all elements of $C^{\ell+1}$ are computed using Algorithms 8. In line 4, we remove all links from

Algorithm 8 GetAllManifolds(G, V, ℓ, j)

Input: $G = (N, E), V \subset E, \ell \in \{0, 1, 2\}$

Output: $S \subset E$ \triangleleft set of link representing all manifolds
1: $E_\ell \leftarrow \{e^k \in E : k = \ell\}$ \triangleleft links of index ℓ
2: $S \leftarrow \emptyset$ \triangleleft initialize S
3: **for all** $c^p \in C^{\ell+1-j}$ **do** \triangleleft for each critical node of index $\ell + 1 - j$
4: $E_\ell \leftarrow E_\ell \setminus S$ \triangleleft remove all already visited links
5: $S \leftarrow S \cup \text{AlternatingRestrictedBFS}(G, V, E_\ell, c^p)$ \triangleleft add current manifold to S

the set of admissible links that are already visited by previous integration steps. Since this is also done in Algorithm 2, each link is traversed only once.

Given the set of links S describing the manifolds of all critical nodes of index $\ell + 1 - j$, we compute the intersection $I \subset E$ with respect to the critical nodes in the boundary of S using Algorithm 4. The intersection I contains the links of the combi-

Algorithm 9 CountPaths(G, V, I, c^p, ℓ)

Input: $G = (N, E), V \subset E, I \subset E, c^p \in N$

Output: $C_c \subset N$ \triangleleft adjacent critical nodes w.r.t. \mathbb{Z}_2
1: $N_S \leftarrow \text{GetManifoldNodes}(G, V, I, c^p, \ell)$ \triangleleft nodes covered by 1-separatrices of c^p
2: $C_{N_S} \leftarrow \{u^p \in N_S : \nexists \{u^p, w^k\} \in V\}$ \triangleleft critical nodes in N_S
3: $P \leftarrow \emptyset$ \triangleleft control container for the breadth-first search
4: $L \leftarrow \{c^p\}$ \triangleleft initialize list of visited nodes with c^p
5: $Q.\text{push}(\{c^p, \text{false}\})$ \triangleleft all links of c^p are unmatched
6: **while** $Q \neq \emptyset$ **do** \triangleleft constrained breadth-first search
7: $\{u^p, \text{flag}\} \leftarrow Q.\text{pop}()$ \triangleleft get the next node and the flag of its links
8: $P \leftarrow P \cup \{u^p\}$ \triangleleft add the node to the control container
9: $W \leftarrow \text{AlternatingEdges}(G, V, u^p, \text{flag})$ \triangleleft get the correctly flagged links
10: $W \leftarrow W \cap I$ \triangleleft consider only links that are also part of the intersection I
11: **for all** $\{u^p, w^k\} \in W$ **do** \triangleleft for each link = {start node, end node}
12: **if** $w^k \in N_S$ **then** \triangleleft the end node must be covered by the 1-separatrices of c^p
13: **if** $u^p \in L$ **then** \triangleleft if the start node is flagged
14: $L \leftarrow L \triangle \{w^k\}$ \triangleleft flag the end node w.r.t. \mathbb{Z}_2
15: $Z \leftarrow \text{AlternatingEdges}(G, V, w^k, \text{flag})$ \triangleleft get the links of the end node
16: $Z \leftarrow Z \cap I$ \triangleleft restrict them to the intersection I
17: $N_Z \leftarrow \{z^q \in N : \exists \{z^q, w^k\} \in Z\}$ \triangleleft get their start nodes
18: **if** $N_Z \subset P$ **then** \triangleleft all start nodes must already be processed
19: $Q.\text{push}(\{w^k, \neg \text{flag}\})$ \triangleleft the breadth-first search can uniquely continue
20: $C_c \leftarrow L \cap C_{N_S} \setminus c^p$ \triangleleft restrict visited nodes to the critical nodes except c^p

natorial streamlines between all $c^{\ell+1}$ and all critical boundary nodes. To compute the boundary of an individual $c^{\ell+1}$, we need to count the number of combinatorial streamlines connecting $c^{\ell+1}$ with c^ℓ .

This is done using a simple graph algorithm shown in Algorithm 9. Its input is the cell graph G , the combinatorial gradient field V , the global intersection I , a critical node c^p , and the layer of integration ℓ .

Since I represents all paths between critical nodes, we need to extract the set of links that describe the local intersection defined by c^p . The nodes covered by these links restrict the counting of the paths. This is obtained in line 1 using Algorithm 10. To count the number of paths between the critical nodes, we apply a constrained breadth-first search (line 6) on the local intersection (line 12).

The constraint is given as follows: If there are multiple paths in the local intersection entering a single node w^k , all paths must be processed before the breadth-first search can continue at w^k (lines 8, and 15-19). This guarantees that there are no racing-conditions between the multiple paths in the breadth-first search.

To get the number of the paths between two critical nodes, we count how often each node is visited during the breadth-first search. Since we are only interested in the Morse-Smale complex with coefficients in \mathbb{Z}_2 , it suffices to count the number of visits modulo 2. This is obtained by taking the symmetric difference \triangle in lines 13 and 14. Finally, we restrict the visited nodes L to the critical nodes (line 20). Those critical nodes are connected by an odd number of separatrices.

Algorithm 10 computes the nodes N_S covered by a combinatorial manifold S that emanates at a critical node c^p . The integration is thereby restricted to the global intersection I (line 2). The nodes covered by the links are uniquely added to N_S (lines 4-8).

Algorithm 10 GetManifoldNodes(G, V, I, c^p, ℓ)

Input: $G = (N, E), V \subset E, c^p \in N, \ell \in \{0, 1, 2\}$

Output: $N_S \subset N$ \triangleleft nodes covered by the integrated manifold

```

1:  $E_\ell \leftarrow \{e^k \in E : k = \ell\}$   $\triangleleft$  links of index  $\ell$ 
2:  $S \leftarrow \text{AlternatingRestrictedBFS}(G, V, E_\ell \cap I, c^p)$   $\triangleleft$  integration restricted to  $I$ 
3:  $N_S \leftarrow \emptyset$   $\triangleleft$  initialize set of nodes
4: for all  $\{u^p, w^k\} \in S$  do  $\triangleleft$  for each link = {start node, end node}
5:   if  $u^p \notin N_S$  then  $\triangleleft$  start node was not yet added
6:      $N_S \leftarrow N_S \cup \{u^p\}$   $\triangleleft$  add start node
7:   if  $w^k \notin N_S$  then  $\triangleleft$  end node was not yet added
8:      $N_S \leftarrow N_S \cup \{w^k\}$   $\triangleleft$  add end node

```

3.4.3 Computational Complexity

We now give a brief analysis of the computational complexity to compute the Morse-Smale complex. We denote the number of vertices of C by n and the number of critical nodes in a combinatorial gradient field by c . For the complexity analysis, we assume a cubical complex is given. The argumentation below, however, can be generalized to simplicial complexes taking the degree of the nodes N in G into account.

Since the Morse-Smale complex can only be computed for a given combinatorial gradient field, we also consider the construction of this field in the complexity analysis. The complexity for the construction of such a combinatorial gradient field is $O(n)$ using

Algorithm 5 – for each node of index 0 we only work on its lower star which has a constant size in the case of cubical complexes.

Analyzing the complexity of the Morse-Smale complex extraction described in Algorithm 7 is more intricate. We start with the essential Algorithm 2: the restricted breadth-first search. Due to line 6, the union in line 8 is disjoint. This implies that the complexity of Algorithm 2 is $O(|T|)$ with T denoting its output (a set of links). Algorithm 8 is used to integrate all manifolds. Since Algorithm 8 only calls Algorithm 2, its complexity is $O(|S|)$ where S denotes the set of links representing the manifolds. The computation of the intersection I using Algorithm 4 has a complexity of $O(|I|)$: due to line 4, the union in line 5 is disjoint.

Finally, we need count the number of paths for each critical node in Algorithm 7. The complexity of the loop (line 3) is $O(c)$ with c denoting the number of critical nodes. The complexity of its body is given by the complexity of Algorithm 9, i.e., the computational effort for the path counting.

To compute the nodes N_S covered by the local intersection (line 1), Algorithm 10 is used. This algorithm is a direct application of the restricted breadth-first search. Each node is uniquely inserted and its complexity is therefore $O(|N_S|) \subseteq O(|I|)$. The complexity of the body of the *while* loop in line 6 of Algorithm 9 is constant. It therefore suffices to count the number of times that line 19 is executed. The node w^k is only inserted into Q if it belongs to the local intersection (line 12) and all neighboring nodes $z^q \in N_Z$ have already been processed (line 8 and 18). Since w^k can only be inserted by a neighboring node z^q , it can therefore be inserted only once. The complexity of Algorithm 9 is hence $O(|I|)$, since only nodes contained in I can enter the queue at all.

In summary, we have the following complexities:

Algorithm 5	$O(n)$
Algorithm 7	
└ Algorithm 8	$O(S)$
Algorithm 4	$O(I)$
while-loop	$O(c)$
└ Algorithm 9	$O(I)$

Since there holds $O(|I|) \subseteq O(|S|) \subseteq O(n)$, the overall complexity of Algorithm 7 is $O(n + |S| + |I| + c|I|) \subseteq O(cn)$. Since there is a lower bound on the worst-case complexity for the Morse-Smale complex extraction problem in three dimensions of $O(n^2)$ [RWS11], our proposed algorithm is optimal.

3.4.4 Implementational Details

Running time To compute the combinatorial gradient field, the cell graph is decomposed into lower stars of the 0-nodes. Since this is a disjoint decomposition, each lower star can be processed in parallel. Also, the boundaries of the critical nodes are independent of each other, which allows a parallel computation. We process Algorithm 4, 7, and 8 in parallel using OpenMP.

We now discuss the influence of the integration direction j on the running time of Algorithm 7. The flag $j \in \{0, 1\}$ of Algorithm 7 influences solely its running time, and not its complexity. In three dimensions, the combinatorial 0-streamlines can only merge, while the 2-streamlines can only split. Due to this property, the computation of the co-boundaries of all 0-nodes ($j = 1, \ell = 0$) and the boundaries of the 3-nodes ($j = 0, \ell = 2$) can be done with a complexity of $O(n)$ [RWS11]. Changing the role of j , on the other hand, yields a worst case complexity of $O(n^2)$: n saddles can be connected

to a single extremum and each of the separation lines is space filling. The computational complexity for the boundary matrices ∂_1 and ∂_3 is therefore $O(n)$. However, the computation of ∂_2 has a worst case complexity of $O(n^2)$, regardless of j [RWS11]. In summary, the flag j yields the following individual complexities

	∂_1	∂_2	∂_3
$j = 0$	$O(n^2)$	$O(n^2)$	$O(n)$
$j = 1$	$O(n)$	$O(n^2)$	$O(n^2)$

Hence, the overall complexity of Algorithm 7 of $O(n^2)$ is not influenced by the choice of $j \in \{0, 1\}$.

In contrast, the practical running time depends on j . For most inputs, the best choice in our experiments was

$$\begin{aligned} \partial_1: & \quad (j = 0, \ell = 0), \\ \partial_2: & \quad (j = 0, \ell = 1), \\ \partial_3: & \quad (j = 1, \ell = 2). \end{aligned}$$

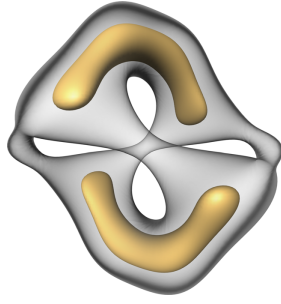
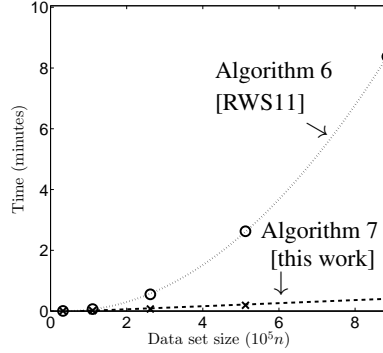
Although the worst-case complexity is $O(n^2)$ for ∂_1 and ∂_3 , in many practical cases the computation of the (co-)boundaries of the 2- and 1-nodes amounts only to a line integration, as in this setting, $|W| \leq 1$ in Algorithm 2, line 5. Therefore, it is beneficial to do a line integration starting at the saddles instead of a volume integration starting at the extrema.

Memory requirements We only need to compute the boundary matrices $\partial_{\ell+1}$ of the Morse-Smale complex C^M . Since $\partial_{\ell+1}$ can be represented as a sparse binary matrix, it does not require much memory. The memory consumption of the cell graph G , on the other hand, depends on its representation. As discussed in Section 3.2.1, a simplicial complex is explicitly and a cubical complex implicitly represented. While the adjacency information of an explicit representation is given by a sparse matrix, it is always computed on-the-fly using index calculations for the implicit representation.

Since we enumerate the nodes N and the links E without gaps, we can represent the combinatorial gradient field V simply by an array of bits of length $|E|$. The sets used in the algorithms of this section can also be represented using Boolean arrays. However, Boolean arrays of size $|E|$ would result in a huge memory overhead when using parallel computation. Since we only work on the intersection of manifolds, arrays of size $O(|I|)$ are sufficient. A look-up map translates global into local indices. This allows for efficient set operations.

If the data values at the 0-cells of the complex are defined by 32-bit single precision floats, then the total memory overhead factor of our method is about 3 in our current implementation for a cubical complex. This enables the analysis of large 3-dimensional image data on commodity hardware.

In case of a simplicial complex, we also need the Boolean masks discussed above. However, we also need to represent the cell graph explicitly. All cells of the complex, i.e., the vertices, edges, faces, and tetrahedrals, are explicitly represented by nodes and their adjacency is represented by a sparse matrix. Depending on the sparsity of the adjacency matrix, this explicit representation might result in a huge memory consumption; the storage of the Boolean masks is negligible.

(a) Analytic function g .

(b) Running times

Figure 3.7: Comparison of running times for the analytic function g . a) The gray and yellow surfaces depict two different isolevels of the analytic function g . b) The circle and cross markers show the running times over the number of vertices n to construct the boundary matrix over different resolutions for Algorithm 6 and Algorithm 7, respectively. The semi-solid line depicts a least-square fitting of a linear function for the cross markers. The dotted line depicts a fitting of a quadratic function for the circle markers.

3.4.5 Performance Comparison

In the following, we compare the performance of Algorithm 6 [RWS11] and our Algorithm 7. The experiments were performed on a machine with two Intel Xeon E5645 CPUs. To compare the performances, both algorithms were run single threaded.

Figure 3.7 depicts an analytic test function. The data set is given by sampling the function $g : [-2, 2]^3 \rightarrow \mathbb{R}$

$$\begin{aligned}
 g(x, y, z) = & 1 \sin(1x) \sin(2y) \sin(3z) + 2 \sin(2x) \sin(1y) \sin(3z) \\
 & + 3 \sin(3x) \sin(2y) \sin(1z) + 4 \sin(1x) \sin(3y) \sin(2z) \\
 & + 5 \sin(2x) \sin(3y) \sin(1z) + 6 \sin(3x) \sin(1y) \sin(2z) \\
 & + 1 \cos(3x) \cos(1y) \cos(2z) + 2 \cos(2x) \cos(1y) \cos(3z) \\
 & + 3 \cos(1x) \cos(2y) \cos(3z) + 4 \cos(3x) \cos(2y) \cos(1z) \\
 & + 5 \cos(2x) \cos(3y) \cos(1z) + 6 \cos(1x) \cos(3y) \cos(2z).
 \end{aligned} \tag{3.1}$$

on a uniform grid of increasing resolution. Figure 3.7a shows an illustration of g . We added a small amount of uniform noise in the range of $[-0.5, 0.5]$ to the samples to investigate the performance of the algorithms.

Figure 3.7b shows the running times of the Morse-Smale complex extraction step using Algorithm 6 by Robins et al. [RWS11] as well as of our Algorithm 7. The running time of Algorithm 6 increases quadratically with the number of vertices in the complex. In contrast, our method shows linear running time.

The running times using a single core applied to the analytic function g sampled on a uniform 96^3 grid are: 486 seconds for Algorithm 6 and 25 seconds for Algorithm 7. The construction of the combinatorial gradient field needs 10 seconds for both cases.

The reason for this behavior stems from the structure of the data which is also often found in real-world data: the function g contains large scale structures but the

added noise results in many critical nodes and the combinatorial 1-streamlines often merge and split. This property dramatically increases the practical running time of the algorithm by Robins et al. [RWS11]. The breadth-first search in Algorithm 6 traverses the links multiple times. On the other hand, our Algorithm 7 is not affected by this perturbation of the data. Each link is only traversed once. This restricted traversal originates the optimal complexity of our Algorithm 7.

3.5 A Hierarchy of Combinatorial Gradient Fields

In real-world data, the Morse-Smale complex might be very rich structured. Especially small-scale structures introduced by noise in the data acquisition process may create spurious topological features. Therefore, one is interested in a meaningful but consistent simplification of the Morse-Smale complex. Our framework allows for this by computing a sequence of combinatorial gradient fields that represents the input field in a hierarchical manner. The user is then able to select an appropriate level of detail to efficiently analyze the data.

The construction of a hierarchy of combinatorial gradient fields consists of two steps. In the first step, the initial combinatorial gradient field $V_0 \subset E$ is computed using Algorithm 5. This gradient field represents the fine-grained combinatorial flow of the input data. In the second step, the combinatorial gradient field is iteratively simplified by removing the smallest fluctuation in every iteration. This simplification is done by computing the ℓ -separation line that represents the smallest height-difference of adjacent critical nodes in a given gradient field V_k . Applying the simplification iteratively yields a sequence \mathcal{V} of combinatorial gradient fields

$$\mathcal{V} = (V_k)_{k=0,\dots,m}. \quad (3.2)$$

We now interpret the gradient fields again as Morse matchings. In graph theoretical terms, an ℓ -separation line $s \subset E$ connecting two critical points is an augmenting path since it is alternating and its start- and end-node are not matched. We can produce a larger matching $V_{k+1} \subset E$ by taking the symmetric difference

$$V_{k+1} = V_k \triangle s. \quad (3.3)$$

Equation (3.3) is called *augmenting* the matching. Since the incident critical nodes of s are matched after the augmentation, the number of critical nodes is decreased by two. An illustration of the augmentation process is given in Figure 3.8. The simplification stops if the matching cannot be augmented anymore. This is the case when the minimal number of critical nodes of a cell complex is reached (Section 2.2), or all remaining augmentations are not valid. This validity constraint is given by Theorem 6 and discussed later in this section. The final result represents the gradient field with the coarsest level of detail. We want to emphasize that an augmentation of a matching only poses local changes in V : there are always exactly two critical points involved and no new critical points are created.

The sequence \mathcal{V} is uniquely defined by the final matching V_m and the augmenting paths $(p_i)_{i=1,\dots,m}$. An arbitrary element $V_k \in \mathcal{V}$ can be restored by

$$V_k = V_m \triangle p_m \triangle \dots \triangle p_{m-k+1}. \quad (3.4)$$

Because the augmenting paths of cell graphs are typically rather short, this is a lot more memory-efficient in our context than storing all elements of \mathcal{V} individually.

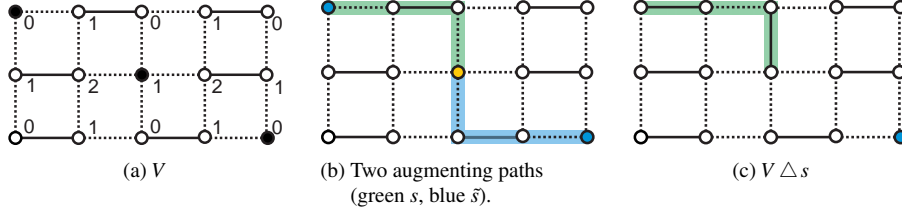


Figure 3.8: Illustration of Equation (3.3). The nodes of the graph G are labeled by their index. The critical points of the combinatorial gradient field V are shown as black spheres. Links belonging to V are depicted as solid black lines. The augmentation of V along the green path s results in a coarser gradient field $V \triangle s$ where the saddle (yellow) at one of its minimum (blue) are not critical anymore. A single minimum (blue dot) remains in $V \triangle s$.

Equation (3.4) enables the user to select a combinatorial gradient field with a prescribed number of critical points. However, there is a lower bound of critical points due to the Morse inequalities, see Section 2.2. Alternatively, we can make use of the height difference of the critical nodes connected by an augmenting path p as an importance measure ω_p . The user can set a fraction $\theta \in [0, 1]$ to select a combinatorial gradient field with a weight as close as possible to $\omega_{p_1} + (1 - \theta)(\omega_{p_m} - \omega_{p_1})$. This approach can be useful in dealing with noisy data. Noise induces a very complex structured Morse-Smale complex. The augmenting paths corresponding to the spurious structures, however, have typically a very small weight, i.e., the two connected critical nodes hardly differ in their scalar value. Hence, setting θ to a small value therefore removes all spurious structures in the Morse-Smale complex while the dominant structures remain unchanged.

The above importance measure ω_p is closely related to the concept of *persistence* [ELZ02]. In two dimensions, both measures yield the same hierarchy [DLL⁺10]. However, this is not the case anymore in three dimensions [Bau11]. Nevertheless, this relationship motivates our use of the heuristic to take the smallest height difference as simplification criteria. In Chapter 4, we introduce the concept of persistence, and discuss its relationship to our heuristic and the problems that occur in higher dimensions in Section 4.4.

We need to make sure that the operation (3.3) does not create any closed combinatorial streamlines. Such streamlines can only occur in the general vector field context, and not in gradient fields. The requirements for the operation (3.3) are given by Forman's cancellation theorem [For98a]:

Theorem 6 (Cancellation Theorem) *If two unmatched nodes are connected by a unique ℓ -separation line s in a Morse matching V , then $V \triangle s$ is a Morse matching.*

Hence, we need to compute the ℓ -separation line that represents the currently smallest height difference and additionally connects two critical nodes uniquely.

In the following, we present the algorithms for an efficient simplification of a given combinatorial gradient field given on a 2- and 3-dimensional cell graph. In two dimensions, the simplification process consists of finding the saddle-extremum pair that has the smallest height difference of all pairs, and is uniquely connected. In the 3-dimensional case, we additionally need to consider the saddle-saddle pairs. From the

algorithmic point of view, the saddle-saddle pairs are much more intricate. A saddle can be connected to an arbitrary number of other saddles. An efficient treatment of these pairs is therefore necessary.

Joswig and Pfetsch [JP06] showed that the computation of a combinatorial gradient field that contains the minimal number of critical nodes is *NP*-hard, in general. While under certain conditions, an optimal simplification is possible in the 2-dimensional case, it fails in three dimensions due to the specific structure of the cell graph. In Section 3.5.5, we discuss this and stress some further properties of the simplification process.

3.5.1 Simplification in 2 Dimensions

As we have seen in Section 3.2.4, the 0- and 1-separation lines in a 2-dimensional cell graph $G = (N, E)$ can either merge or split. This simplifies the construction of the sequence \mathcal{V} . We can integrate the separation lines directly using a restricted breadth-first search.

Algorithm 11 Sequence2D(G, V, \tilde{f})

Input: $G = (N, E), V \subset E, \tilde{f}: N \rightarrow \mathbb{R}$

Output: list of augmenting paths $p, V_m \subset E$

```

1:  $p \leftarrow \emptyset, h \leftarrow \emptyset$   $\triangleleft$  initialize heap  $h$  and list of augmenting paths  $p$   $\triangleleft$  the hierarchy
2:  $\partial_1 \leftarrow \text{BoundaryMatrix}(G, V, 0, 0)$   $\triangleleft$  compute  $\partial_1$ 
3:  $\partial_2 \leftarrow \text{BoundaryMatrix}(G, V, 1, 1)$   $\triangleleft$  compute  $\partial_2$ 
4: for all  $c^1 \in C^1$  do  $\triangleleft$  for each saddle
5:    $\{c_k^0\} \leftarrow \partial_1(c^1)$   $\triangleleft$  get the boundary of the saddle
6:    $\omega_1 \leftarrow \min_{u^0 \in \{c_k^0\}} (\tilde{f}(c^1) - \tilde{f}(u^0))$   $\triangleleft$  compute the smallest height difference
7:    $h.\text{push}(c^1, \omega_1)$   $\triangleleft$  insert the saddle with the height difference in the heap
8:    $\{c_k^2\} \leftarrow \partial_2^{-1}(c^1)$   $\triangleleft$  get the coboundary of the saddle
9:    $\omega_2 \leftarrow \min_{u^2 \in \{c_k^2\}} (\tilde{f}(u^2) - \tilde{f}(c^1))$   $\triangleleft$  compute the smallest height difference
10:   $h.\text{push}(c^1, \omega_2)$   $\triangleleft$  insert the saddle with the height difference in the heap
11: while  $h \neq \emptyset$  do  $\triangleleft$  simplification process
12:   $(c^1, \omega) \leftarrow H.\text{pop}()$   $\triangleleft$  get the next saddle with its weight
13:  if  $c^1 \in C^1$  then  $\triangleleft$  check whether the saddle is still critical
14:     $(s, \omega) \leftarrow \text{GetMinWeight2D}(G, V, c^1)$   $\triangleleft$  currently smallest height difference
15:     $(\tilde{c}^1, \tilde{\omega}) \leftarrow h.\text{top}()$   $\triangleleft$  next element in the heap
16:    if  $\omega \leq \tilde{\omega}$  then  $\triangleleft$  is the current weight smaller than the next one
17:       $V \leftarrow V \triangleleft s$   $\triangleleft$  augment the matching
18:       $p.\text{push}(s)$   $\triangleleft$  save the augmenting path
19:    else  $\triangleleft$  defer the saddle
20:       $h.\text{push}(c^1, \omega)$   $\triangleleft$  reinsert the saddle with the new weight
21:  $V_m \leftarrow V$   $\triangleleft$  final matching

```

The pseudo-code for constructing the sequence \mathcal{V} is given Algorithm 11. The input is the cell graph G , the combinatorial gradient field V computed using Algorithm 5, and the extended input function \tilde{f} (Section 3.3). The output consists of V_m – the coarse-grained gradient field – and a list of augmenting paths p . Together, these can be used to reconstruct an arbitrary element of the sequence \mathcal{V} (3.2).

In the first step, the boundary matrices ∂_1 and ∂_2 are computed using Algorithm 7 based on the given gradient field V (line 2 and 3). From the (co)boundary information

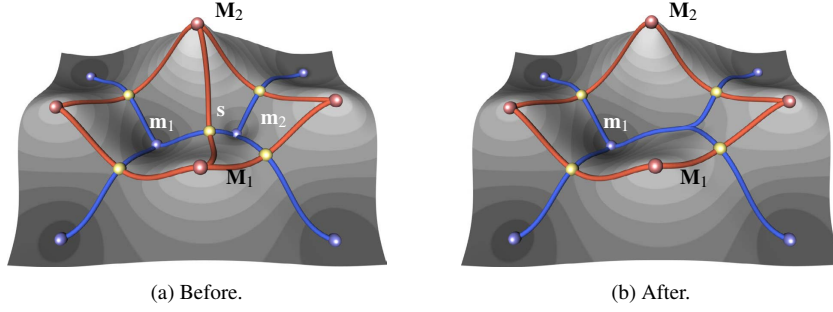


Figure 3.9: Cancellation of the saddle-minimum pair (s, m_2) . The minima, saddles, and maxima are depicted as blue, yellow, and red spheres, respectively. The 0- and 1-separation lines are shown as blue and red lines.

of all saddles $c^1 \in C^1$ (lines 5 and 8), we can choose the adjacent minimum and maximum with the smallest height difference (line 6 and 9). We use the height difference as weight ω for the saddle c^1 . We add c^1 together with its weight ω in a heap structure h (line 7 and 10). Based on those weights, we can detect the currently smallest fluctuation. The heap h is organized in a way such that the element with the smallest weight is the top element.

In the second step, we construct the sequence \mathcal{V} (line 11). The top element of the heap h is a saddle together with its weight. However, this weight can be outdated. The augmentation process (3.3) changes the boundary information and thereby the height differences. Critical points that were not connected in a prior level can be connected now, or even non-critical anymore. Therefore, we need to check whether c^1 is still a saddle (line 13). If this is the case, we compute the currently smallest height difference ω of the saddle c^1 to its adjacent extrema (line 14) using Algorithm 12. If ω is still smaller than the weight of the next element (lines 15 and 16), we can augment the current gradient field V by taking the symmetric difference \triangle of V and the separation line s representing this weight (line 17). We store the path s in a container p to recover this level of detail (line 18). Note that the saddle c^1 and the extremum are not critical anymore after applying the symmetric difference. We update derived information as the sets of critical points C^k iteratively during an augmentation of V . On the other hand, if the weight ω is larger than the weight of the next element (line 19), we reinsert the saddle c^1 with its new weight ω (line 20). If there are no valid augmenting paths, the process stops and the gradient field V becomes the final element of the sequence \mathcal{V} (line 21).

The implication of the above augmentation for the Morse-Smale complex is shown in Figure 3.9a. Let us consider the saddle-minimum pair (s, m_2) . After an augmentation along the separatrix connecting (s, m_2) , these two nodes are not critical anymore, see Figure 3.9b. The connectivity of the adjacent critical nodes has also changed. All saddles that were connected to m_2 are now connected to m_1 . The separatrices which connected s_1 to the maxima M_1 and M_2 are removed from the complex.

Algorithm 12 computes the separation line s with the smallest weight ω for a given saddle c^1 . For a saddle c^1 , we first compute the 0-separation line s_1 with the smallest weight ω_1 (line 1) and then the 1-separation line s_2 with the smallest weight ω_2 (line 2) using Algorithm 13. The separation line with the smallest weight (line 3) is used as an augmenting path (line 4 and 6).

Algorithm 12 GetMinWeight2D(G, V, \tilde{f}, c^1)**Input:** $G = (N, E), V \subset E, \tilde{f} : N \rightarrow \mathbb{R}, c^1 \in C^1$

Output: $s \subset E, \omega \in \mathbb{R}$ \triangleleft augmenting path and its weight
 1: $(s_1, \omega_1) \leftarrow \text{GetPath2D}(G, V, \tilde{f}, c^1, 0)$ \triangleleft best path to the minima
 2: $(s_2, \omega_2) \leftarrow \text{GetPath2D}(G, V, \tilde{f}, c^1, 1)$ \triangleleft best path to the maxima
 3: **if** $\omega_1 \leq \omega_2$ **then**
 4: $(s, \omega) \leftarrow (s_1, \omega_1)$ \triangleleft best pair is a saddle-minimum pair
 5: **else**
 6: $(s, \omega) \leftarrow (s_2, \omega_2)$ \triangleleft best pair is a saddle-maximum pair

The computation of a j -separation line s and its weight ω is done in Algorithm 13. The separation line s of a saddle c^1 is part of its descending ($j = 0$) or ascending manifold ($j = 1$). In the first step, we therefore compute the descending/ascending manifold S of c^1 (line 2). The critical nodes covered by the links S are the connected extrema N_S (line 3). However, we need to be careful with the virtual 1-separation lines (Section 3.2.4). Those end in a non-critical boundary cell of the cell complex (line 3) and are no valid augmenting paths. If both separatrices end in the same extremum, an augmentation along one them would create a closed streamline in the augmented matching, which can not occur in combinatorial gradient fields (Theorem 6). Therefore, we ignore the extrema that are twice connected to c^1 (line 4). This also correspond to the coefficient group \mathbb{Z}_2 . The j -separation line s of the extremum representing the smallest height difference is returned (line 5 and 6).

Algorithm 13 GetPath2D(G, V, \tilde{f}, c^1, j)**Input:** $G = (N, E), V \subset E, \tilde{f} : N \rightarrow \mathbb{R}, c^1 \in C^1, j \in \{0, 1\}$

Output: $s \subset E, \omega \in \mathbb{R}$ \triangleleft path and its weight
 1: $q \leftarrow 2j, s \leftarrow \emptyset, \omega \leftarrow \infty$ \triangleleft initialize
 2: $S \leftarrow \text{GetManifold}(G, V, c^1, j)$ \triangleleft integrate both separation lines
 3: $N_S \leftarrow (N(S) \cap C^q) \cup \{v^1 \in N(S) : |\{v^1, w^2\} \in E\}| = 1\}$ \triangleleft get the end-nodes
 4: **if** $|N_S| > 1$ & $N_S \cap C^q \neq \emptyset$ **then** \triangleleft two different end-nodes, at least one is critical
 5: $(u, \omega) \leftarrow \min_{\tilde{u} \in N_S \cap C^q} |\tilde{f}(c^1) - \tilde{f}(\tilde{u})|$ \triangleleft smallest height difference (critical only)
 6: $s \leftarrow \text{AlternatingRestrictedBFS}(G, V, S, u)$ \triangleleft get the sole separation line

3.5.2 2D Examples

We first illustrate the ability to extract the Morse-Smale complex of a scalar field where noise is present. We then apply Algorithm 11 to several triangulated 2D manifolds provided by Aim@Shape [Aim]. All examples were computed on a laptop containing an Intel Core i7-2720QM CPU.

For an illustration of a sequence \mathcal{V} , we applied Algorithm 11 to a synthetic data set depicted as a height field in Figure 3.10. The data set was produced by sampling the analytic function $f : [-1, 1]^2 \rightarrow \mathbb{R}$

$$f(x, y) = \sin(10x) \sin(10y) e^{-3(x^2+y^2)} \quad (3.5)$$

on a uniform triangulation with $16k$ vertices. We then added uniform noise in the the range of $[-0.05, 0.05]$ to the subdomain $[0, 1] \times [-1, 1]$. The running time for the

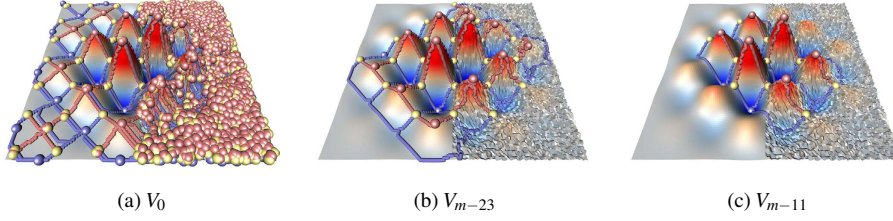


Figure 3.10: Synthetic noisy scalar field. Minima, saddles and maxima are depicted as blue, yellow and red spheres, while 0- and 1-separation lines are shown as blue and red lines, respectively.

computation of \mathcal{V} using Algorithm 11 was less than a second on a standard laptop. Figure 3.10 shows the Morse-Smale complex of the initial combinatorial gradient field V_0 , and two elements V_{m-23} and V_{m-11} of the sequence \mathcal{V} . As can be seen in Figure 3.10a, V_0 includes all structures induced by the noise. The simplified combinatorial gradient fields, however, only contain the dominant critical points and separation lines present in f .

The overall running time for the computation of (3.2) using Algorithm 11 applied to the Gaussian curvature of several surface models is shown in Table 3.4. The worst case complexity of Algorithm 11 is $O(n^3)$, where n denotes the number of edges in the triangulation. However, the empirical running time for practical applications is almost linear (Table 3.4).

Surface Model	triangles	nodes in G	links in G	time (sec)
screwdriver	54300	162902	325800	<1
dinosaur	112384	337154	674304	<1
knot	957408	2872224	5744448	7

Table 3.4: Running time of Algorithm 11.

3.5.3 Simplification in 3 Dimensions

We now describe the construction of the sequence \mathcal{V} for a given combinatorial gradient field V on a 3-dimensional cell graph $G = (N, E)$. The main idea is similar to the 2-dimensional case. We want to compute the separation line between two critical points that represents the currently smallest height difference. However, the specific graph structure of G in three dimensions challenges this computation. While the 'best' extremum-saddle pair can be similarly computed as in Section 3.5.1, the efficient computation of the 'best' saddle-saddle pair is more intricate. In prior work by Lewiner [Lew05], this resulted in a non-feasible running time, and by Gyualssy [Gyu08], it induced a large memory consumption (Section 1.2). Algorithm 14 shows a memory and running time efficient alternative. An illustration of it is given in Figure 3.11.

Algorithm 14 depicts the pseudo-code to compute the sequence \mathcal{V} . The main structure is very similar to the 2-dimensional case: we construct the heap containing all saddles with their weight, then, the simplification process starts. However, we need to take care about the specific structure of the cell graph in three dimensions. There are

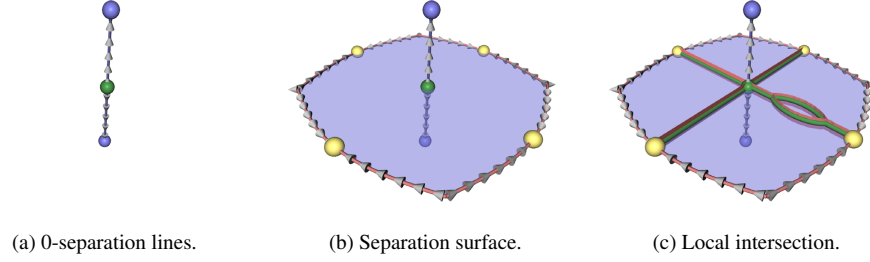


Figure 3.11: Sketch of Algorithm 14. The descending manifolds of the 1-saddle (green sphere) are the 0-separation lines that connect the saddle to the minima (blue sphere). Firstly, the minimum-saddle pair with the smallest height difference is chosen. Secondly, the local intersection defined by the 1-saddle is computed. The ascending manifolds (blue surface) is integrated in a preprocessing step. This separation surface ends in 2-separation lines (red lines) that emanate from 2-saddles (yellow spheres). The 1-separation lines that connect the 1-saddle with the 2-saddles are shown as green lines. The right 2-saddle is connected twice with the 1-saddle. This saddle is therefore not a valid candidate for a simplification. From the remaining 2-saddles and the two minima the critical node is chosen that has the smallest weight with respect to the 1-saddle.

two types of saddles and the 1-separation lines can merge and split. Therefore, we need to apply some modifications.

The heap construction is done by Algorithm 15 (line 2). In line 8, we compute the current weight of a saddle $c \in C^1 \cup C^2$ using Algorithm 16. This also involves the computation of the 1-separation lines, which can merge and split. However, the sets of links covered by these lines are nested with respect to the sequence \mathcal{V} . Less and less links get covered but no new links are added applying Equation (3.3) iteratively. We can make use of this property by precomputing the set of links I describing the 1-

Algorithm 14 Sequence3D(G, V, \tilde{f})

Input: $G = (N, E), V \subset E, \tilde{f} : N \rightarrow \mathbb{R}$

Output: list of augmenting paths $p, V_m \subset E$

1:	$p \leftarrow \emptyset$	◁ the hierarchy
2:	$h \leftarrow \text{InitHeap}(G, V, \tilde{f})$	◁ initialize list of augmenting paths
3:	$S \leftarrow \text{GetAllManifolds}(G, V, 1, 0)$	◁ construct the heap
4:	$I \leftarrow \text{GetIntersection}(G, V, S, 1, 0)$	◁ integrate all descending separation surfaces
5:	while $h \neq \emptyset$ do	◁ intersect them with the ascending surfaces
6:	$(c, \omega) \leftarrow H.\text{pop}()$	◁ simplification process
7:	if $c \in C^1 \cup C^2$ then	◁ get the next saddle with its weight
8:	$(s, \omega) \leftarrow \text{GetMinWeight3D}(G, V, I, \tilde{f}, c)$	◁ is the node still critical
9:	$(\tilde{c}, \tilde{\omega}) \leftarrow h.\text{top}()$	◁ smallest height difference
10:	if $\omega \leq \tilde{\omega}$ then	◁ get the next saddle with its weight
11:	$V \leftarrow V \triangle s$	◁ is the current weight smaller than the next one
12:	$p.\text{push}(s)$	◁ augment the matching
13:	else	◁ save the augmenting path
14:	$h.\text{push}(c, \omega)$	◁ defer the saddle
15:	$V_m \leftarrow V$	◁ reinsert the saddle with its new weight
		◁ final matching

separation lines in the initial gradient field V_0 (lines 3 and 4) and use them as restriction in the upcoming breadth-first searches. This reduces the traversals in the graph G to a minimum.

Algorithm 15 InitHeap(G, V, \tilde{f})

Input: $G = (N, E), V \subset E, \tilde{f}: N \rightarrow \mathbb{R}$

Output: heap h

		◁ the heap (priority queue)
1:	$h \leftarrow \emptyset$	◁ initialize
2:	$\partial_1 \leftarrow \text{BoundaryMatrix}(G, V, 0, 0)$	◁ compute ∂_1
3:	$\partial_2 \leftarrow \text{BoundaryMatrix}(G, V, 1, 1)$	◁ compute ∂_2
4:	$\partial_3 \leftarrow \text{BoundaryMatrix}(G, V, 1, 2)$	◁ compute ∂_3
5:	for all $c \in C^1 \cup C^2$ do	◁ for each 1- and 2-saddle
6:	$\{c_k^0\} \leftarrow \partial_1(c)$	◁ get 1-boundary of c (minima)
7:	$\omega_0 \leftarrow \min_{u^0 \in \{c_k^0\}} (\tilde{f}(c) - \tilde{f}(u^0))$	◁ compute the smallest height difference
8:	$h.\text{push}(c, \omega_0)$	◁ insert the saddle with the height difference in the heap
9:	$\{c_k^1\} \leftarrow \partial_2(c)$	◁ get 2-boundary of c (1-saddles)
10:	$\omega_1 \leftarrow \min_{u^1 \in \{c_k^1\}} (\tilde{f}(c) - \tilde{f}(u^1))$	◁ compute the smallest height difference
11:	$h.\text{push}(c, \omega_1)$	◁ insert the saddle with the height difference in the heap
12:	$\{c_k^3\} \leftarrow \partial_3^{-1}(c)$	◁ get 2-coboundary of c (maxima)
13:	$\omega_3 \leftarrow \min_{u^3 \in \{c_k^3\}} (\tilde{f}(u^3) - \tilde{f}(c))$	◁ compute the smallest height difference
14:	$h.\text{push}(c, \omega_3)$	◁ insert the saddle with the height difference in the heap

The construction of the heap h is done using Algorithm 15. We first compute the boundary matrices ∂_1 (line 2), ∂_2 (line 3), and ∂_3 (line 4). For each saddle $c \in C^1 \cup C^2$, we compute the adjacent neighboring critical points and push the saddle c with its minimal height difference as weight ω in the heap h (lines 6-11). Note that c is inserted multiple times in h . However, we always check in Algorithm 14 if the saddle was already processed (line 5).

Algorithm 16 computes the separation line s with the smallest weight ω for a given saddle $c \in C^1 \cup C^2$ that allows for an augmentation of the combinatorial gradient field. To do so, we first compute the separation lines connecting c to its adjacent extrema using Algorithm 17. In case $c \in C^1$, the 0-separation line s_0 representing the smallest height difference ω_0 is computed (line 1), and, in case $c \in C^2$, the corresponding 2-separation line s_2 and its weight ω_2 (line 2). Note that these two cases are disjunct due to the layer of integration. The separation line is used as a preliminary augmenting path (lines 3-6) unless we find a 1-separation line with a smaller weight.

The computation of the 1-separation line s_1 with the smallest weight ω_1 starts at line 7. Firstly, the local intersection S is extracted from the global I (line 7). Given this set of links, all adjacent saddles $\{\tilde{c}_i\}$ of c that are covered by S are collected (line 8). Note that this set can be empty. For each of the adjacent saddles $\{\tilde{c}_i\}$, we compute their weights $\{\tilde{\omega}_i\}$ as the height difference of c and \tilde{c}_i (lines 9 and 10). We sort the saddles in ascending order (line 11). In case there is an adjacent saddle \tilde{c}_i with a smaller weight than the saddle-extremum pair (line 13), we check if the connection between the two saddles c and \tilde{c}_i is unique using Algorithm 18 (line 14). If the connection is unique, we have found the augmenting path s with the smallest weight ω (lines 15 and 16). Otherwise we continue the iteration (line 12).

Algorithm 17 computes the 0- or 2-separation lines that connect a saddle c to its adjacent extrema. This algorithm resembles the 2-dimensional analogue given in Al-

Algorithm 16 GetMinWeight3D(G, V, I, \tilde{f}, c)

Input: $G = (N, E), V \subset E, I \subset E, \tilde{f} : N \rightarrow \mathbb{R}, c \in C^1 \cup C^2$

Output: $s \subset E, \omega \in \mathbb{R}$ \triangleleft augmenting path and its weight

- 1: $(s_0, \omega_0) \leftarrow \text{GetPath3D}(G, V, \tilde{f}, c, 0)$ \triangleleft best path to the minima
- 2: $(s_2, \omega_2) \leftarrow \text{GetPath3D}(G, V, \tilde{f}, c, 2)$ \triangleleft best path to the maxima
- 3: **if** ω_0 **then** \triangleleft c is a 1-saddle
- 4: $(s, \omega) \leftarrow (s_0, \omega_0)$ \triangleleft take the 0-separation line
- 5: **else** \triangleleft c is a 2-saddle
- 6: $(s, \omega) \leftarrow (s_2, \omega_2)$ \triangleleft take the 2-separation line
- 7: $S \leftarrow \text{AlternatingRestrictedBFS}(G, V, I, c)$ \triangleleft integrate local intersection
- 8: $\{\tilde{c}_i\} \leftarrow N(S) \cap (C^1 \cup C^2) \setminus \{c\}$ \triangleleft get boundary critical nodes (saddles)
- 9: **for** $\tilde{u} \in \{\tilde{c}_i\}$ **do** \triangleleft for each saddle of opposite index
- 10: $\tilde{\omega}_i \leftarrow |\tilde{f}(c) - \tilde{f}(\tilde{u})|$ \triangleleft compute height difference
- 11: $(\tilde{c}_i, \tilde{\omega}_i) \leftarrow \text{sort}(\{\tilde{c}_i, \tilde{\omega}_i\}, \text{ascending})$ \triangleleft sort saddles in ascending order
- 12: **for** $(\tilde{u}, \tilde{\omega}) \in (\tilde{c}_i, \tilde{\omega}_i)$ **do** \triangleleft for each saddle of opposite index
- 13: **if** $\tilde{\omega} \leq \omega$ **then** \triangleleft is the weight smaller than the one of the extremum
- 14: $(s_1, \text{flag}) \leftarrow \text{IsUnique}(G, V, S, \tilde{u})$ \triangleleft compute connections
- 15: **if** $\text{flag} = \text{true}$ **then** \triangleleft is there a single connection between the two saddles
- 16: $(s, \omega) \leftarrow (s_1, \tilde{\omega})$ \triangleleft use this saddle-pair as simplification pair

gorithm 13. The separation line s of a saddle c is part of its descending ($j = 0$) or ascending manifold ($j = 2$). In the first step, we compute the descending/ascending manifold S of c (line 2). The critical nodes covered by the links S are the connected extrema N_S (line 3). A restricted breadth-first search applied to the extremum with the smallest height difference (line 5) yields the corresponding separation line s . Note that the 2-separation lines can end in non-critical bounding cells (line 3) similar as in the 2-dimensional case. In this case, they are virtual and no valid augmenting path; we ignore these lines.

The unique 1-separation line that connects two saddles is computed in Algorithm 18. Here we use a modified version of the restricted breadth-first search given in Algorithm 2. The basic idea is to start a combinatorial streamline integration from a saddle \tilde{c} (line 2). The integration needs to be alternating with respect to V (line 7) but we restrict it to the already integrated local set of 1-separation lines S (line 8). By design, all 1-separation lines emanating from \tilde{c} restricted to S must end in the saddle c . Hence, we know there is more than one path connecting \tilde{c} and c if we observe a split of a 1-separation line. We can abort the integration in this case (lines 12-14). If the separation

Algorithm 17 GetPath3D(G, V, \tilde{f}, c, j)

Input: $G = (N, E), V \subset E, \tilde{f} : N \rightarrow \mathbb{R}, c \in C^1 \cup C^2, j \in \{0, 2\}$

Output: $s \subset E, \omega \in \mathbb{R}$ \triangleleft path and its weight

- 1: $q \leftarrow 1.5j, s \leftarrow \emptyset, \omega \leftarrow \infty$ \triangleleft initialize
- 2: $S \leftarrow \text{GetManifold}(G, V, c, j)$ \triangleleft integrate the two separation lines
- 3: $N_S \leftarrow (N(S) \cap C^q) \cup \{v^2 \in N(S) : |\{v^2, w^3\} \in E\}| = 1\}$ \triangleleft get the end-nodes
- 4: **if** $|N_S| > 1$ & $N_S \cap C^q \neq \emptyset$ **then** \triangleleft two different end-nodes, at least one is critical
- 5: $(u, \omega) \leftarrow \min_{\tilde{u} \in N_S \cap C^q} |\tilde{f}(c^1) - \tilde{f}(\tilde{u})|$ \triangleleft smallest height difference, critical only
- 6: $s \leftarrow \text{AlternatingRestrictedBFS}(G, V, S, u)$ \triangleleft get the sole separation line

line is unique, there exists only a sole link to continue the integration (lines 9-11). The integration stops if we get to a critical point – the saddle c (line 5).

Algorithm 18 IsUnique(G, V, S, \tilde{c})

Input: $G = (N, E), V \subset E, S \subset E, \tilde{c} \in C^1 \cup C^2$

Output: $s \subset E, \text{unique} \in \{\text{true}, \text{false}\}$ \triangleleft unique augmenting path

```

1:  $s \leftarrow \emptyset, \text{unique} \leftarrow \text{true}, Q \leftarrow \emptyset$   $\triangleleft$  initialize
2:  $Q.\text{push}(\{\tilde{c}, \text{false}\})$   $\triangleleft$  integration starts with  $\tilde{c}$ , all incident links are unmatched
3: while  $Q \neq \emptyset$  do  $\triangleleft$  breadth-first search
4:    $\{u, \text{flag}\} \leftarrow Q.\text{pop}()$   $\triangleleft$  next node and the corresponding link flag
5:   if  $u \in (C^1 \cup C^2) \setminus \{\tilde{c}\}$  then  $\triangleleft$  we are done, found the other saddle
6:     return
7:    $W \leftarrow \text{AlternatingEdges}(G, V, u, \text{flag})$   $\triangleleft$  get the next links for integration
8:    $W \leftarrow (W \cap S) \setminus s$   $\triangleleft$  restriction to local intersection, remove already visited links
9:   if  $|W| = 1$  then  $\triangleleft$  integration can be uniquely continued
10:     $s \leftarrow s \cup \{u, v\} \in W$   $\triangleleft$  add link to augmenting path
11:     $Q.\text{push}(\{v, \neg \text{flag}\})$   $\triangleleft$  push next node with negated link flag
12:   else  $\triangleleft$  found a split of 1-separation lines
13:      $\text{unique} \leftarrow \text{false}$   $\triangleleft$  no unique connection possible
14:   return  $\triangleleft$  abort

```

3.5.4 Performance Analysis

In the following, we provide a performance analysis for the computation of the sequence \mathcal{V} including a complexity analysis for practical cases. The running time and the memory consumption of our algorithm are measured for different kinds of data sets. All measurements were done on an Intel Xeon E5530 2.4 GHz system.

Applying Algorithm 14 to different data sets, we observed that noise significantly increased the running time. Therefore, we consider a random field as a practical worst case. We measured the behavior of the simplification algorithm with increasing data size for two different scalar fields. The first one is a uniform random field of size 256^3 with a range $[-0.5, 0.5]$, from which we took nested subfields of dimensions 32^3 , 64^3 , 128^3 and 256^3 . For each of the subfields, we computed the complete sequence \mathcal{V} using Algorithm 14. The second field is a signed distance field of a microporous structure. In the same way as for the random field, we have chosen a nested sequence of subfields. Figure 3.12 shows in logarithmic scaling the running time and memory consumption for both sequences.

Computational Complexity

The computational complexity heavily depends on the topological complexity of the input data: the number of critical points and their connectivity. Both are a priori not known. Especially, the 1-separation lines are crucial. Such lines could be space filling, i.e., the lines cover the entire combinatorial separation surfaces which themselves can cover the entire domain, and their computation is no longer of the complexity of a line-integration but of a volume-integration. The design of a function with such kind of space-filling curves is a non-trivial task. However, most of the input data are well defined avoiding such behavior. In the following, we provide an analysis of Algorithm 14

for typical non-degenerated input data. The number of vertices of the complex X is denoted by n .

Neglecting degenerated cases, the complexity should be at most $O(n^{5/3})$; in practice it seems to be lower.

Algorithm 14 initially computes all combinatorial separation surfaces of the 2-saddles (line 3) avoiding multiple traversals of the links. A reasonable upper bound for the size of a surface is $n^{2/3}$. Since the integration is done for all 2-saddles, the computation of the combinatorial surfaces is of complexity $O(|C^2|n^{2/3}) \subset O(n^{5/3})$ with $|C^2|$ denoting the number of 2-saddles.

The computation of the 1-separation lines is restricted to the separation surfaces S , and each link in S is only traversed once (Algorithm 4, line 4). According to Section 3.4.3, the complexity of Algorithm 4 is $O(|I|) \subset O(|S|) \subset O(n^{2/3})$.

The crucial part in Algorithm 14 is the *while*-loop (line 5). The heap h consists of all saddles. Each saddle is inserted with its 'best' minimum, opposite saddle, and maximum. Hence, the size of h is bounded by *const* c with c denoting the number of 1- and 2-saddles. In line 10, the weight of the current saddle is checked whether it is the currently smallest weight. In the worst case, all saddles are deferred, and every time we need to iterate over the complete heap h to find the pair with the smallest height difference. In practical cases, however, only *const* c -operations are needed to do so.

In Algorithm 16, the pair that represents the smallest height difference is computed. Algorithm 17 computes the 0- or 2-separation line connecting the saddle to the extrema. In practical cases, those lines are not space filling and their integration is a line-integration of complexity $O(n^{1/3})$.

The computation of the 1-separation lines, on the other hand, is more intricate. The local intersection is extracted from the global set of links I . Considering non-degenerated cases, the separation surfaces are not space filling. However, noise in the input data may create surface-filling 1-separation lines that often merge and split. An upper bound is therefore $n^{2/3}$. Each link is only traversed once (Algorithm 18). Therefore, the practical complexity of Algorithm 16 is $O(n^{2/3})$.

In summary, the practical complexity to compute the hierarchy is: $O(|S| + |I| + cn^{2/3}) \subset O(n^{5/3} + n + cn^{2/3}) \subset O(n^{5/3})$. We want to note that the theoretical worst-case complexity is $O(n^3)$ due to the possible saddle-deferring in the *while*-loop.

While we observed the practical complexity for the artificial random field, we observed almost linear running time for the distance function (see Figure 3.12a). This indicates that the 1-separation lines are well distributed on well-defined data.

Memory Consumption

The following analysis of the memory consumption is given for a cubical complex, i.e., the degree of the nodes N in G is constant.

For the construction of the sequence, a Boolean vector, whose size is given by the number of links in G , is needed to represent the current matching. The number of nodes in the cell graph is eight times the number of vertices in the input grid. The number of links in G is therefore bounded by 24 times the number of vertices. Since the size of a Boolean is 1/32th of a single precision number, we need a factor of 0.75 of the input data to represent the matching. Three additional Boolean vectors of size $|N|$ are needed: the surface integration, its intersection, and the node matching. The total factor is therefore 1.5 of the input data.

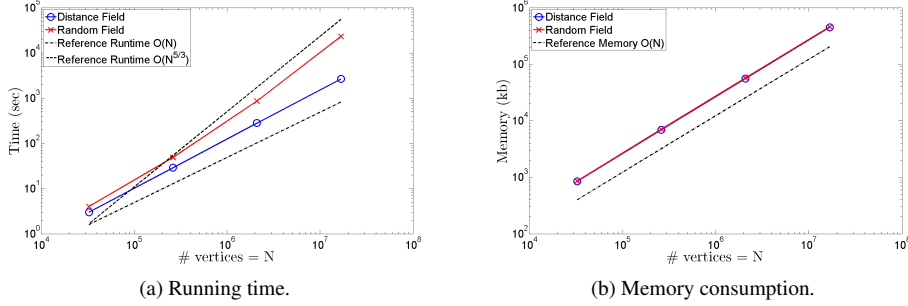


Figure 3.12: Performance of Algorithm 14. The red solid lines show the running time and the memory consumption for a uniform random field, the blue solid line for a distance field. The axes are shown in a loglog-scale. The full sequence \mathcal{V} was measured in all cases. The black dotted lines indicate a complexity of $O(n)$ and $O(n^{5/3})$.

Robins et al. [RWS11] proved that the critical points are in a 1-1 correspondence to the topological changes in the lower level sets. Since (3.3) only decreases the number of critical nodes, the size of the heap is given by the number of critical nodes in the input field. In practice, there are much fewer critical nodes than regular nodes. Hence, the size of the h is negligible. Note that only the scalar values for these critical points need to be stored. They define the weight of an augmenting path.

The theoretical maximal memory consumption for separation surfaces is bounded by the number of 1- and 2-nodes in G . Space filling surfaces, however, do not appear in practice, and this bound is much lower, in general. Hence, the relative amount of memory needed by Algorithm 14 is constant which is substantiated by the results shown in Figure 3.12b.

3.5.5 Properties of the Hierarchization

In the following, we discuss three properties of the hierarchization process. These properties especially occur in the 3-dimensional case.

Optimality. The overall goal of Section 3.5 is the computation of a sequence \mathcal{V} such that the topological complexity – the number of critical nodes – within \mathcal{V} is decreasing and the last element $V_m \in \mathcal{V}$ contains the minimal number of critical nodes based on the Morse inequalities (2.6). While this is possible in polynomial time for 2-dimensional complexes that are manifold-like as shown by Lewiner et al. [LLT03], the general problem turns out to be *NP*-complete. This result was proven by Joswig and Pfetsch [JP06]:

Theorem 7 *Given a simplicial complex X and a nonnegative integer k , it is strongly *NP*-complete to decide whether there exists a Morse matching with at most k critical faces, even if X is connected, pure, 2-dimensional, and can be embedded in \mathbb{R}^3 .*

Algorithm 14 therefore ends with a non-optimal Morse matching, in general. In practice, this results in the fact that no unique augmenting paths can be found anymore. Therefore, the final matching V_m contains more critical nodes than the Morse inequalities demand, in general.

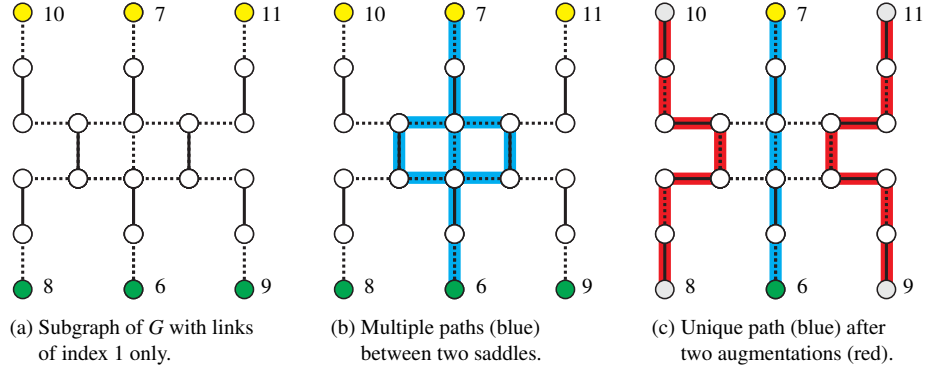


Figure 3.13: Depiction of a combinatorial gradient field in a subgraph of G connecting 2-saddles (yellow) and 1-saddles (green). The critical nodes are labeled with their assigned scalar value. The links of the matching are depicted as black solid lines.

Monotony. The construction of the sequence \mathcal{V} is based on the heuristic that we want to remove the currently smallest fluctuation in the data. The pair of critical nodes that is connected by the augmenting path p_i should have the smallest height difference of all possible pairs. Applying the simplification process iteratively, the height difference of critical nodes should therefore increase monotonically, and, hence, the weights of the augmenting paths as well. While this is the case for the 2-dimensional case, this cannot be expected in a 3-dimensional cell graph due to its specific structure, as explained in the following.

Due to the degree of the 1- and 2-nodes, the 1-separation lines can merge and split. Consider a subgraph of G with different saddle-saddle pairs as shown in Figure 3.13a. Assume that the central pair represents the smallest height difference ($\omega = 1$) and that the yellow saddle popped out first from the heap h in Algorithm 14. However, this pair cannot be augmented since there are three paths connecting them, see Figure 3.13b. The yellow saddle is deferred with a greater weight given by its adjacent extrema. In the next steps, other saddle pairs are augmented with a greater weight ($\omega = 2$), see Figure 3.13c. Parts of the corresponding augmenting paths, however, share links with

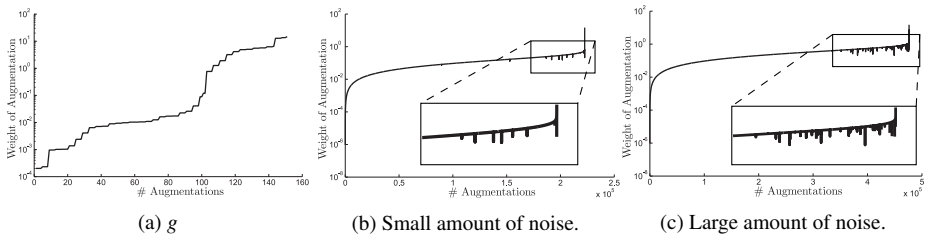


Figure 3.14: Graph of the weights of the augmenting paths. (a) shows the weights of augmentations for the artificial function g (3.1) over the number of augmentations. The weights behave monotonically increasing. The monotony is broken in (b) where a small amount of noise is added to g . The number of monotony breaks further increases if the level of noise is increased (c).

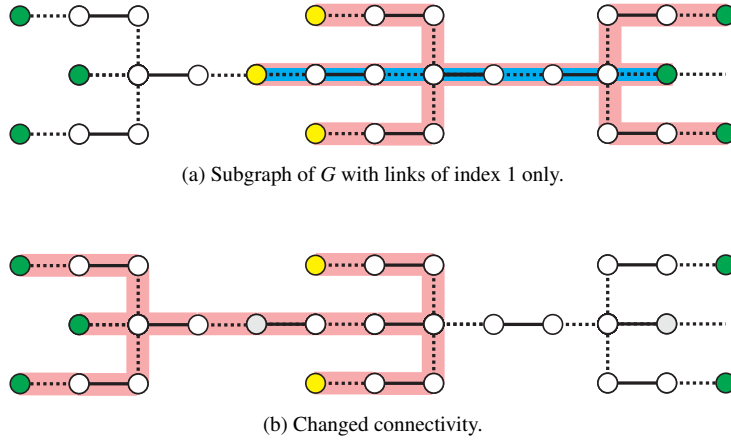


Figure 3.15: Illustration of the connectivity change due to a saddle-saddle simplification. Shown is a subgraph of G connecting 2-saddles (yellow) and 1-saddles (green). The links of the matching are depicted as black solid lines. The blue line in (a) depicts an augmenting path. After the augmentation, the connectivity of the saddles (red lines) completely changed (b).

the paths connecting the central pair. Due to the augmentations, the orientation of the links is changed, and the central pair is suddenly uniquely connected. If the central yellow saddle is now next in line, the breadth-first search in Algorithm 18 determines the central green saddle as partner with the smallest height difference ($\omega = 1$). Since the connection between them is now unique, the path allows for an augmentation. However, the weight of the augmentation is smaller than in the prior operations.

In practice, the breaks of monotony in the weight sequence are caused by noise. The perturbations introduced by it force the 1-separation lines to short split-merge sequences as depicted in Figure 3.13b. To analyze this behavior, we sampled the artificial function g (3.1) on an 128^3 grid and added two levels of uniform noise to it. Figure 3.14 shows the results. The 1-separation lines in the pure function g are well distributed. No deferring of saddles in the above sense can be observed. The weights of the augmenting paths are monotonically increasing, see Figure 3.14a. However, adding a small amount of noise in the range of $[-0.5, 0.5]$ results in 12 monotony breaks, see Figure 3.14b. If we add noise in the range of $[-1, 1]$, the number of breaks increases further to 59, see Figure 3.14c. This indicates that the level of noise heavily influences the number of splits in the 1-separation lines.

Given an element V_k of \mathcal{V} , this element might contain spurious pairs of critical nodes, i.e., pairs that have a height difference smaller than the chosen minimal height. However, investigating the graph of the augmentation weights as shown in Figure 3.14 gives an impression about the strength of the breaks and which level might be appropriate.

Connectivity. A simplification of a saddle-saddle pair can completely change the connectivity of the adjacent saddles. Figure 3.15 depicts an exemplarily situation. It may happen that the 1-separation lines of multiple 2-saddles merge and share several links before they split again and end in 1-saddles, see Figure 3.15a. The 1-streamline described by the shared links is in some sense a narrow one-way street.

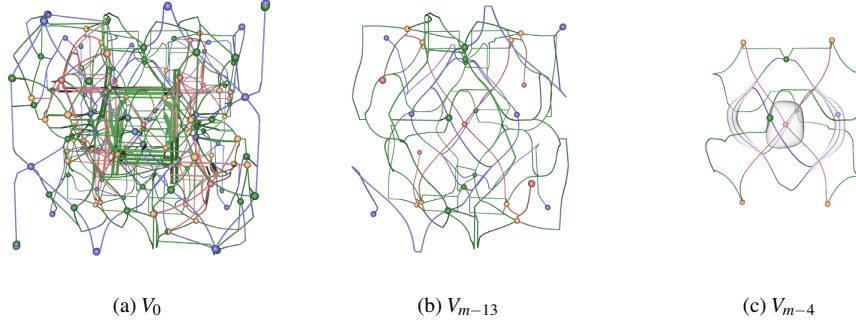


Figure 3.16: Illustration of different levels of details of \mathcal{V} of the analytic function g (3.1). Minima, 1-saddles, 2-saddles and maxima are depicted as blue, green, yellow and red spheres, respectively. The p -separation lines are shown as blue ($p = 0$), green ($p = 1$) and red ($p = 2$) lines. The isosurface (grey) in (c) illustrates the most dominant minima and maxima regions. The hierarchy consists of 207 levels.

All 1-streamlines on the left side of this one-way street must cross it to enter the right side¹.

However, simplifying the gradient field along one of the 1-separation lines (the blue line in Figure 3.15a) changes the connectivity of the saddles. Before the simplification, all 2-saddles (yellow) were connected to the 1-saddles (green) on the right side. Only the central 2-saddle was also connected to the 1-saddles on the left side. After the simplification, none of the remaining 2-saddles is connected to the right 1-saddles anymore, see Figure 3.15b. There is no 1-separation line connecting them. All of them are now connected to the 1-saddles on the left side, which were not in their boundary before, see Figure 3.15b.

Due to this property, it is necessary to recompute the (co)boundary information of a saddle when it comes out of the heap. Previous simplification steps may have changed its connectivity – even if this saddle was not directly involved. An explicit storage of the connectivity of the critical points (as in [Gyu08]) is therefore also algorithmic challenging. After every simplification step, the boundary information of all critical points must be updated: new pairs are created but old pairs must also be removed. In our graph theoretical setting, this is implicitly done. Specific data structures are therefore not necessary. We want to note that this behavior cannot occur in the saddle-extremum case due to the specific cell structure (Section 3.2.4). Saddles can only be added to the (co)boundary of an extremum during an augmentation; a removal is only possible if they take part in a simplification.

3.5.6 3D Examples

In the following, we present some examples to illustrate the running time Algorithm 14. The experiments were done on an Intel Core i7-2720QM CPU with 16 GB RAM. Table 3.5 shows the running time for different 3-dimensional data sets of varying dimensions and topological complexity. Besides computing the complete sequence \mathcal{V} , it is in some cases sufficient to compute only a subsequence of \mathcal{V} in order to remove only the most

¹This also indicates why the overall complexity for the computation of the Morse-Smale complex is quadratic. In the worst case, all 2-saddles are connected to all 1-saddles.

Data Set (Size)	Neghip (64 ³)	Hydrogen (128 ³)	Aneurism (256 ³)	Beetle (416 ² × 247)	Benzene (401 ³)	Synthetic (1024 ³)
$\mathcal{V}_{5\%}$ (Size)	3s 2'804	26s 12'103	5m 01s 27'516	8m 03s 236'317	10m 20s 87	173m 50s 167
\mathcal{V} (Size)	3s 2'852	27s 12'107	7m 04s 37'476	8m 45s 248'243	10m 22s 118	174m 50s 207

Table 3.5: Running times of Algorithm 14. The first and second rows show the running times for the computation as well as the number of levels of a 5% and a complete simplification, respectively.

spurious/noisy topological structures. Therefore, we also give the computation time of Algorithm 14 for a 5% simplification, i.e., until the weight of the last augmenting path corresponds to 5% of the data range. The size of the sequence, i.e., the number of levels of detail, is given as well. Figure 3.16 shows the separation lines for different levels of detail of the synthetic example g , see Equation (3.1). The running time is also given in Table 3.5.

The construction time of \mathcal{V} for the complex aneurism data set was approximately 7 minutes, which correlates to the work of Gyulassy et al. [GBPH11] with a reasonable valence parameter. The aneurism shows also that the topological complexity of the input data influences the running time of Algorithm 14. While for simple data sets as the neghip, the hydrogen, or even the stagbeetle almost no difference can be observed in the running times between the subsequence $\mathcal{V}_{5\%}$ and \mathcal{V} , the running time increases by 40% for the aneurism.

Chapter 4

Quantification of Critical Points

In the previous chapter, we presented algorithms to compute a sequence \mathcal{V} of combinatorial gradient fields and an explicit representation of the implicitly encoded Morse-Smale complex. The sequence also allows for a differentiation of spurious and dominant critical points. However, the computation of the sequence is based on the assumption that two critical points are connected by a unique separation line. As shown by Joswig and Pfetscher [JP06], not all critical points can be paired in this way. In fact, this is an NP-complete task in Forman’s combinatorial setting, in general.

Edelsbrunner et al. [ELZ02] introduced the concept of persistent homology (Section 4.1). In contrast to the homotopic operations in Forman’s combinatorial setting, this theory investigates the homological changes of the lower-level sets. In particular, persistent homology has drawn much attention since it robustly extracts the topological structure of the data.

The main idea of the persistence algorithm is to create the boundary matrices ∂_k of the cell complex C as defined in Section 2.4. A matrix reduction, i.e., a Gaussian elimination, performs a pairing of the cells such that the birth and deaths of homological features are described by pairs of cells. Those cells are exactly the critical points describing the homological changes in the lower-level sets. The strength of these homological features is assessed by the importance measure *persistence*, which can be directly read from the reduced matrix. Similar as the sequence \mathcal{V} , persistence allows for a differentiation of spurious and dominant critical points. However, all critical points are paired¹ in contrast to the homotopic approach described in Section 3.5. This enables a finer differentiation of the critical points.

In the context of computing persistence of 3-dimensional images, which are often noisy due to inaccuracy of the acquisition process, algorithms with good practical running times have been proposed [CK11b] for exact computation, see Section 4.2. However, exact computation of persistence, especially for large 3-dimensional image data, remains a challenging problem due to huge memory requirements.

Forman’s discrete Morse theory [For98b, For01] allows us to reduce data in a way which preserves its topological structure. This representation of the data – the Morse-Smale complex – is much more compact but still contains all the necessary topological information for persistent homology computation.

Inspired by the work of Robins et al. [RWS11], we use discrete Morse theory to compute persistent homology. The strategy introduced in Section 4.3 allows for an

¹Persistence pairs all cell of a cell complex except those representing its topology. However, the concept of *extended persistence* [CSEH09] can be used to also pair those cells.

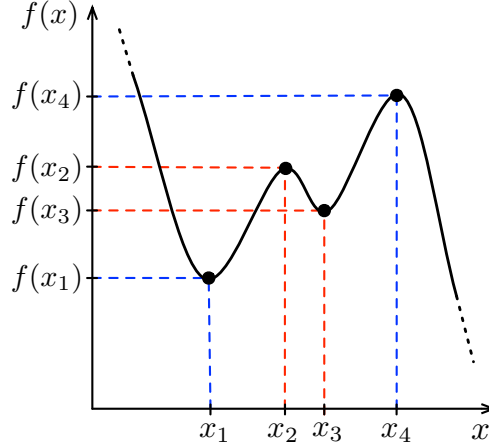


Figure 4.1: Persistent homology of a 1D function $f(x)$. The persistence pairs consist of (x_1, x_4) and (x_3, x_2) . The persistence of x_1 and x_4 is therefore given by $f(x_4) - f(x_1)$, while the persistence of x_2 and x_3 is given by $f(x_2) - f(x_3)$.

output-sensitive computation of persistence with a complexity of $O(cn + c^3)$ with n denoting the size of the input and c the number of critical points within. While the original, algebraic algorithm [ELZ02] needs $O(n^3)$, the currently best persistence algorithm [MMS11] needs $O(n^\omega)$ with $\omega = 2.376$. However, the number of critical points is much lower than the size of the input in many practical cases. Hence, an output-sensitive algorithm is preferable.

Our approach also allows for a memory-efficient computation of persistent homology of large 3-dimensional images. For example, we only need about 14 GB of memory for a data set of size $1120 \times 1131 \times 1552$, in contrast to the 500 GB that would be necessary using standard algebraic algorithms. Additionally, it enables a parallel computation on commodity hardware. The presented strategy is based on the published works [GRWH11, GRWH12].

After describing the computation of persistent homology, we discuss in Section 4.4 the relationship between persistence and the simplification process presented in Section 3.5. Although both concepts may yield the same hierarchy in two dimensions, this is no longer the case in higher dimensions.

Section 4.5 is devoted to an application of persistent homology. We compute correspondences between persistent feature points on near-isometric surfaces. In general, the search space of intrinsic correspondences between two surfaces is too large to be computationally tractable. Therefore, we use persistence to reduce the search space by extracting distinctive features from both surfaces. Such features typically also have the benefit of being more reliable to match because of their distinctiveness. Thus, the search space is reduced in a strategic way.

We present a conceptually direct and simple algorithmic pipeline that is able to match accurately feature points between two near-isometric surfaces. The pipeline consists of established components that enables a straight-forward implementation. This application of persistent homology is based on the work [YGW⁺12].

4.1 Persistent Homology

Before we introduce persistent homology, we want to recapitulate how our input is given. The input function $f : \Omega \rightarrow \mathbb{R}$ only defines scalar values to the 0-cells of the induced cell complex X . As described in Section 3.3, we extend f to all cells by the lower-star filtration: each cell is assigned the maximum function value of the vertices it contains. A *filtration* of the complex X with respect to f is given by the sub-level set $X^t = f^{-1}(-\infty, t]$ with $X^t \subseteq X^s$ for $t \leq s$. Imagine that we start with an empty complex and at each step of the filtration one or more cells are added.

We now give a basic intuition behind homology and persistent homology. We follow here the works of Edelsbrunner et al. [ELZ02, EH10]. For our purposes, we can say that homology detects topological features: connected components, tunnels, and voids for a *fixed* thresholding (sub-level set) of a scalar function f . *Persistent* homology, in turn, describes the *evolution* of topological features looking at consecutive thresholds.

For a 1-dimensional function f as shown in Figure 4.1, this measure can be defined by considering the number of components of the sub-level sets. As t increases the number of components in X^t changes: when t passes the value of a local minimum a component is born, while two components merge when t passes the value of a local maximum. In this case, the maximum is paired with the larger minimum of the two merged components. The persistence of the paired critical points is given as the difference of their function values.

More precisely, given a complex X and a *filtering function* $f : X \rightarrow \mathbb{R}$, *persistent homology* studies homological changes of the sub-level complexes, $X^t = f^{-1}(-\infty, t]$. Persistent homology captures the birth and death times of homology classes of the sub-level complexes, as the threshold t grows from $-\infty$ to $+\infty$. By birth, we mean that a homology feature comes into being; by death, we mean it either becomes trivial or becomes identical to some other class born earlier. The *persistence*, or lifetime of a class, is the difference between its death and birth times. Homology classes with larger persistence reveal information about the global structure of the space X , described by the function f . The persistent homology groups are formally given as follows:

Consider two elements of the filtration: X^t and X^{t+p} . To define the persistent homology groups, we need to factor the k -th cycles Z_k^t of X^t by the k -th boundaries B_k^{t+p} of X^{t+p} . The p -th persistent k -th homology group of a filtration element X^t is given by

$$H_k^{t,p} = Z_k^t / (B_k^{t+p} \cap Z_k^t). \quad (4.1)$$

Loosely speaking, the elements of $H_k^{t,p}$ are those cycles in X^t that do not become boundaries for at least p steps. The rank of $H_k^{t,p}$ denotes the k -th *persistent Betti Number* $\beta_k^{t,p}$ of X^t .

The overall output of the persistence computation is the list of *persistence pairs* of the form (birth, death). This information can be visualized in different ways. One well-accepted idea is the persistence diagram [CSEH07], which is a set of points in a two-dimensional plane, each corresponding to a persistent homology class. The coordinates of such a point are the birth and death times of the related class. An example is given in Figure 4.2b in Section 4.3.1.

An important justification of the usage of persistence is the stability theorem. Cohen-Steiner et al. [CSEH07] proved that for any two filtering functions f and g , the differ-

ence of their persistence is always upperbounded by the L^∞ -norm of their difference:

$$\|f - g\|_\infty := \max_{x \in X} |f(x) - g(x)|. \quad (4.2)$$

This enables robust estimation of how persistence is affected by perturbation of the input. For instance, adding noise to a function changes its persistence diagram only by the noise amplitude. Also, this guarantees that persistence can be used as a signature. Whenever two persistence outputs are essentially different, we know that the functions are definitely different.

4.2 Related Work on Persistent Homology

In the following, we give an overview of previous work on *computing* persistence. For general applications of persistence see [EH10]; for applications in the context of image data, see [BEK10, MW10, RKG⁺11].

The standard, algebraic algorithm [ELZ02, EH10] for persistence has cubic running time in the size of the input. While an example was constructed by Morozov [Mor05], showing that this pessimistic estimation can actually occur, the behavior of this algorithm is only slightly super-linear in practical situations [CK11b].

When focusing on 0-dimensional homology, union-find data structures can be used to compute persistence in time $O(n\alpha(n))$ [EH10], where α is the inverse of the Ackermann functions and n is the input size.

Milosavljevic et al. [MMS11] computed persistent homology in matrix multiplication time $O(n^\omega)$ where the currently best estimation of ω is 2.376. Chen and Kerber [CK11a] proposed a randomized algorithm to compute only pairs with persistence above a chosen threshold. Despite showing better theoretical complexity, it is unclear whether these methods are better than the standard persistence algorithm, in practice.

A recent variation of the standard algebraic algorithm [EH10], called *killing*, introduced by Chen and Kerber [CK11b] significantly reduces the amount of computations. This idea was also used in [WCV12], to compute persistence for n -dimensional images.

In general, purely algebraic methods suffer from high memory requirements. For example, consider a data set of size $1120 \times 1131 \times 1552$. The standard, algebraic persistence algorithms [ELZ02, MMS11, CK11b, WCV12] work on the boundary matrices ∂_k of the cell complex C as introduced in Section 2.1. Since all cells of C are considered, we can compute the memory consumption exactly neglecting boundary effects. Assuming a sparse representation of the boundary matrix $\partial_k : n^k \times n^{k-1} \rightarrow \{0, 1\}$, the amount of its memory is given by

$$\sum_{k=1}^3 8|\partial_k| + 16n^k \quad (4.3)$$

where $|\partial_k|$ denotes the number of matrix entries and n^k the number of k -dimensional cells. Applying formula (4.3) to our example results in about 500 GB of memory, which cannot be processed on commodity hardware. Note that Equation (4.3) does not depend on the topological complexity but only on the data size.

In our approach, we alleviate this effect by reducing the size of data.

4.3 Computation of Persistent Homology

While algorithms with good practical running times have been proposed [CK11b], exact computation of persistence for large 3-dimensional image data remains a challenging problem due to huge memory requirements.

However, Forman proves that the homology of the cell complex C is always isomorphic to the homology of the Morse-Smale complex C^M [For98b]. If the critical nodes contained in the combinatorial gradient field V correspond 1-1 to the topological changes in the sub-level complexes, then the persistent homology of C coincides with the persistent homology of C^M [RWS11].

Lewiner [Lew05] conjectured already that persistence could be efficiently computed using discrete Morse theory. The first algorithmic approach was done by Robins et al. [RWS11]. However, their algorithm to compute the Morse-Smale complex has cubic running time. Inspired by their works, we use discrete Morse theory to compute persistent homology based on the Morse-Smale complex. We greatly benefit herein from the optimal complexity in the Morse-Smale complex computation (Section 3.4.3).

To compute persistence, we first compute the boundary matrices ∂_k using Algorithm 7. As discussed in Section 3.4, the computational complexity of our Algorithm 7 is $O(cn) \subseteq O(n^2)$ with c denoting the number of critical points and n the number of vertices in a cubical complex C .

Then, we use the standard, algebraic algorithm [EH10] with a modification by Chen and Kerber [CK11b]. This algorithm operates on a boundary matrix ∂_k of the cubical complex C representing the input data. The performance modification introduced by Chen and Kerber significantly reduces the amount of computations, in practice. It exploits the fact that cells being the *creators* of homology classes are zeroed in the reduced matrix. By reordering the computations (starting from higher dimensional cells), one first computes the *killer* cells. At this point the associated *creator* is also known, so its column can be zeroed avoiding any computation. A *reduced matrix* is computed, from which the list of *persistent pairs*, as defined in Section 4.1, can be easily read. The modification does not improve the worst-case complexity, which is still $O(n^3)$. But the authors of the paper show that in practical situations this modification reduces running time by an order of magnitude. We refer the reader to [CK11b] for more details.

In contrast to previous work in computing persistence [CK11b, WCV12], we apply the matrix reduction algorithm to the Morse-Smale complex C^M instead of the initial cubical complex C . Since C^M is much smaller than C in typical situations, storing the boundary matrices consumes significantly less memory (see Table 4.1).

The computational complexity for the matrix reduction algorithm, which we use to compute the persistent homology, is $O(c^2 n \log n)$ in the case of cubical complexes. Since we apply it to the Morse-Smale complex, the complexity reduces to $O(c^3)$. It solely depends on the topological complexity of the input data. Including the Morse-Smale complex computation, the complete complexity for our algorithm is therefore $O(cn + c^3)$.

4.3.1 3D Examples

In the following, we present some examples to illustrate our method. All experiments were performed on a machine with two Intel Xeon E5645 CPUs, which provide 12 physical and 24 logical cores, and 24 GB RAM.

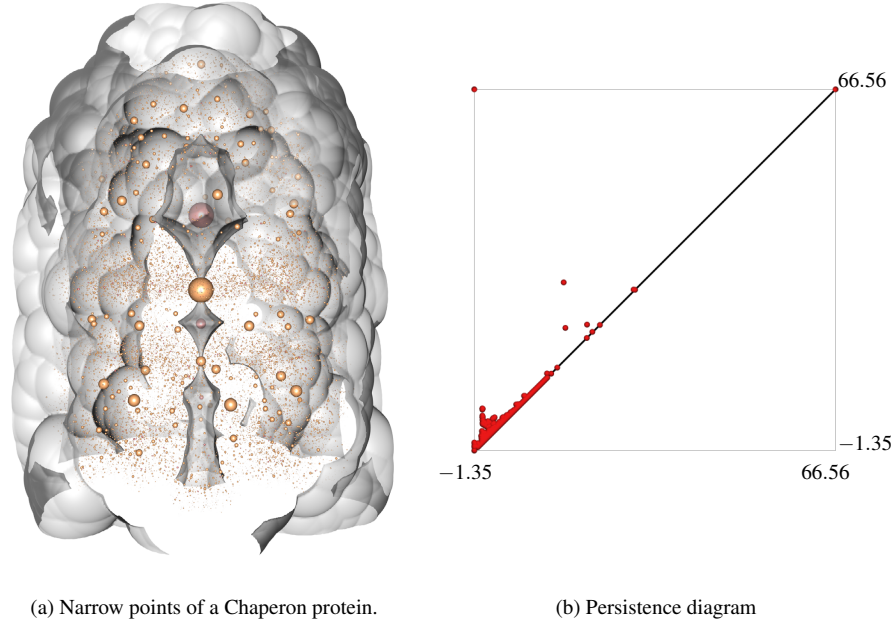


Figure 4.2: Distance field of a Chaperon protein. An isosurface of a distance field, computed from a protein, as gray transparent surface is shown in image a). The maxima and the 2-saddles are shown as red and yellow spheres, respectively. Each sphere is scaled by its persistence. Image b) shows the persistence diagram of the data set containing all dimensions. The axes denote the data range of the distance field.

Table 4.1 shows the running time and memory consumption for different 3-dimensional image data sets provided by [GGK, Bar, Röt]. We measured the total memory usage of our method with one and 24 cores. We also included the memory consumption and running times of a very efficient persistence method [WCV12] working on the boundary matrices of the initial cubical complex. Note that a further comparison to other techniques is also given in [WCV12].

The total memory consumption of our method is up to a factor of 30 less than using the persistence approach of Wagner et al. [WCV12]. In practice, the Morse-Smale complex C^M is much smaller than the cubical complex C . This enables the persistence computation of large data. The memory overhead for the parallel computation is neglectable.

In general, the running times depend on the number of critical points and their connectivity, as can be seen for instance at the Xmas-Present example. In contrast to the other examples, we only observe a speed-up factor of three using a parallel computation. This indicates that the columns in the boundary matrix are very entangled which increases its reduction time. However, using the Morse-Smale complex to compute persistent homology decreases enormously the running time for the Xmas-Present compared to the timings using the complete cell complex. The running times for the other examples are similar to the timings of Wagner et al. [WCV12] using a single threaded computation. However, all examples benefit from a parallel computation. We observe a speedup factor of up to ten in our current implementation using 12 physical cores.

To investigate the behavior of Algorithm 7 for noisy data, we sampled pure uniform noise in the range of $[0, 1]$ on a uniform 256^3 grid. While the cell complex consists of about 10^8 nodes, its gradient field contains only about 10^7 critical points. Even in the case of pure noise, the majority of nodes in G are non-critical nodes, which yields a reasonable memory consumption of a factor 2 less than the approach of Wagner et al. [WCV12]. Due to the large number of critical points and the absence of large scale structures, the corresponding separatrices are relatively small and well distributed. Hence, they can be efficiently integrated. The timings as well as the memory consumption are also given in Table 4.1.

We applied our method to a distance field, computed from a Chaperone protein, shown in Figure 4.2. The objective is the extraction of the maxima and 2-saddles. While the maxima represent the points with the greatest distance to the atoms, the 2-saddles correspond to the narrow points of the field: they define the minimal size of an atom to enter the molecule from the outside. The data set is of dimension $1120 \times 1131 \times 1552$ and contains 1'766'615 critical points. The approach of Wagner et al. [WCV12] computation would theoretically require about 500 GB memory. Our approach, in contrast, only requires about 14 GB even using multiple cores, and can thereby be applied on commodity hardware. The total running time as well as the memory consumption for this example are shown in the last row of Table 4.1.

4.3.2 Discussion

We presented an algorithmic strategy for a memory- and running time-efficient persistent homology computation. Our strategy combines many useful properties:

1. The overall complexity for the persistence computation is $O(cn + c^3)$.
2. The computation of persistence using the Morse-Smale complex requires significantly less memory.

There are some limitations of our approach:

1. Extending our techniques to more general inputs such as simplicial complexes is possible, but would result in high memory-usage – we heavily exploit the compact representation of the initial, cubical complex.
2. Our current method is limited up to three dimensions.
3. Using the single-threaded version, our current implementation can be considerably slower than the algebraic approach for medium-sized data.

Despite these drawbacks, we believe that our method enables the application of persistent homology in new fields. Our current implementation can already be used to analyze very large, complex data sets.

It would be also interesting to see how our proposed algorithm scales for higher dimensional data (see limitation 2). The challenging part is thereby that the combinatorial gradient field may contain extra spurious critical points in contrast to the 3-dimensional case [RWS11].

A fundamental question, which is still an open problem in the literature, is the relation of the topological complexity of a given input data and the persistence computation times (see limitation 3). Since matrix reduction is a global operation, the structure of the underlying Morse-Smale complex is crucial. This structure also depends on the

imaging process and the data format. For instance, the aneurysm and bonsai data are given as 8-bit integer while the prone and supine data are 16-bit integer CT scans. This may also contribute to the different timings shown in Table 4.1.

Table 4.1: Running times and memory consumption for 3D images of different size and topological complexity. The second column shows the topological properties of the data sets. The third column shows the total memory consumption of our method compared to Wagner et al. [WCV12] using 1 and 24 logical cores. The total running time for the construction of the initial gradient field, the boundary matrix and the matrix reduction using 1 and 24 logical cores are compared to the times of Wagner et al. [WCV12] in the fourth column.

Data	Dimensions	Properties		Memory (MB)			Time (sec)		
		$\Sigma C_i $	$ I $	[WCV12]	1 ×	24 ×	[WCV12]	1 ×	24 ×
Silicium	$98 \times 34 \times 34$	1'109	13'882	30	1	1	5	1	< 1
Fuel	$64 \times 64 \times 64$	667	4'982	82	1	1	1	3	< 1
Neghip	$64 \times 64 \times 64$	5'709	47'025	82	2	3	2	3	< 1
Hydrogen	$128 \times 128 \times 128$	24'257	168'626	538	17	19	37	21	2
Engine	$256 \times 256 \times 128$	1'035'127	7'331'167	2'127	296	236	82	141	16
X-mas Present	$246 \times 246 \times 221$	4'836'087	21'201'265	3'112	727	901	16'007	444	153
Aneurysm	$256 \times 256 \times 256$	75'485	1'308'765	4'250	135	146	211	170	14
Bonsai	$256 \times 256 \times 256$	344'277	5'886'696	4'250	225	273	175	219	22
Foot	$256 \times 256 \times 256$	1'658'617	12'162'264	4'250	405	505	171	270	30
Noise	$256 \times 256 \times 256$	11'761'873	42'389'191	*	1'517	1'866	*	619	101
Supine	$512 \times 512 \times 426$	27'440'949	142'886'726	26'133	4'701	5'876	1'496	2'369	339
Prone	$512 \times 512 \times 463$	28'976'885	152'326'748	28'406	5'009	6'262	2'180	2'525	354
X-mas Tree	$512 \times 499 \times 512$	50'043'123	215'181'923	*	7'392	9'162	*	3'374	562
Molecule	$1'120 \times 1131 \times 1552$	1'766'615	40'178'429	*	13'837	14'168	*	18'134	1'504

4.4 Topological Simplification and Persistence

In the following, we briefly discuss the relationship between persistence and topological simplification to create a sequence of combinatorial gradient fields $\mathcal{V} = (V_i)_{i=0\dots m}$ as introduced in Section 3.5. We focus here on the 3-dimensional case.

Topologically simplifying a combinatorial gradient field V means to reduce the number of its critical points in order to create different levels of detail of the input function f . All fine-grained topological features are present in the initial gradient field V , while the last level contains only the dominant topological features of the scalar field f .

Such a hierarchy can be obtained by increasing the set of links in V without introducing any closed p -streamlines as discussed in Section 3.5. Consider two critical points u^p and w^{p+1} in a cell graph $G = (N, E)$ that are connected by a unique p -separation line q . This line is given as a sequence of alternating links that belong either to $E \setminus V$ or V . Taking the symmetric difference $\tilde{V} = V \triangle q$ yields a new combinatorial gradient field \tilde{V} with an increased set of links where u^p and w^{p+1} are not critical anymore, i.e., the links incident to these points are now matched. Note that this operation does not create any cycles as long as there is a unique path connecting the two critical points [For98b].

The symmetric difference not only removes two critical points from a combinatorial gradient field, it also increases the length/area of the affected separatrices. The extremum-saddle simplification (s_b, m) in the top row of Figure 4.3 yields a merge of the separation lines ℓ_1, ℓ_2 with ℓ_a : the length of ℓ_a is increased while ℓ_1 and ℓ_2 are removed. The saddle-saddle simplification (s, s_b) in the lower row of Figure 4.3, on the other hand, increases the area of the separation surface S_a : the surface S_b merges into S_a .

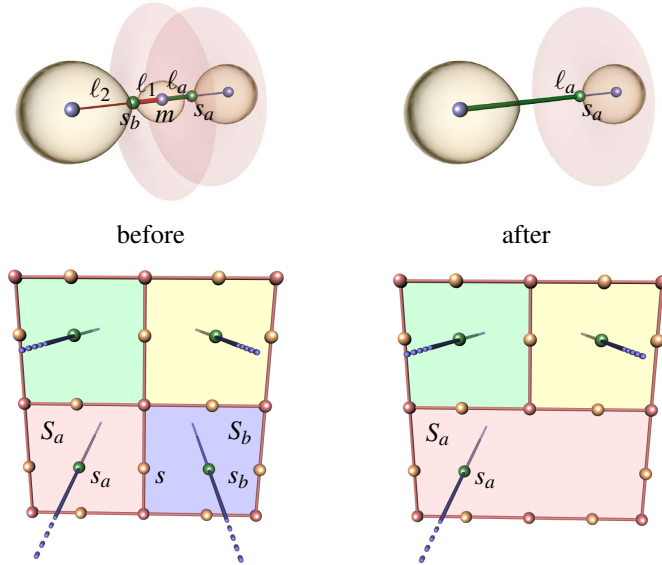


Figure 4.3: Simplification increases the length (top row) and area (bottom row) of affected separation lines and surfaces. Shown are the topological structures before (left) and after (right) a simplification.

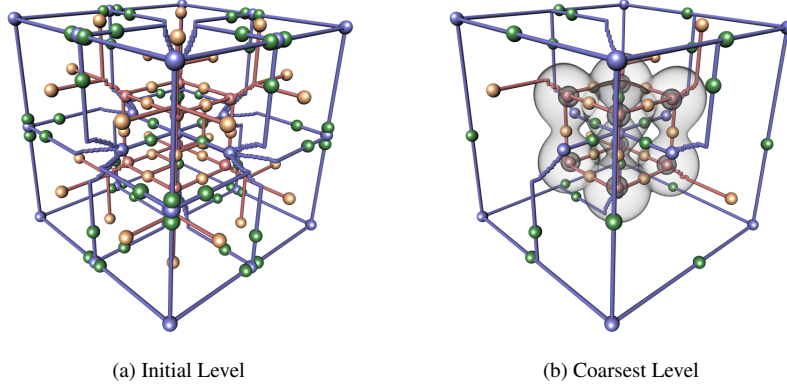


Figure 4.4: Persistence-based simplification. Shown are the extremal structures of the initial field V_0 (left) and the final field V_m (right), which still contains critical points. Gray isosurfaces illustrate the underlying synthetic function.

A hierarchy of combinatorial gradient fields $(V_i)_{i=0\dots m}$ could be obtained by iteratively taking the symmetric difference with respect to the persistence pairs (Section 4.1). However, this symmetric difference does not necessarily yield a combinatorial gradient field where upcoming persistence pairs can be canceled in the sense of Forman, as extensively discussed by Bauer et al. [Bau11, BLW12]: a necessary property to reduce the number of critical points in the sense of Forman is collapsibility of a discrete Morse function, but this is not always given for a generic 3-dimensional Morse function.

The above theoretical argument results, in practice, in the following situation: when creating a hierarchy in the order of the persistence pairs, one arrives rather early in a deadlock where a unique separation line between two paired critical points does not exist. An artificial example is shown in Figure 4.4. The input function consists of 129 critical points. However, only the first 32 persistence pairs could be removed using the symmetric difference. The coarsest representation of the input function still contains 65 critical points. In fact, it follows from Joswig et al. [JP06] that it is an NP-hard problem to pair critical points such that V_m contains the minimal number of critical points, which itself is given by the topology of the domain (for a uniform lattice as shown in Figure 4.4 this is a sole minimum).

However, Dey et al. [DLL⁺10] prove that the pairs of critical points created by the simplification as described in Section 3.5 coincide with the persistence pairs for the special case of a 2-dimensional Morse functions given on a smooth manifold. This relation motivates the greedy approach in Algorithm 14 for the 3-dimensional case.

In practice, we found that the greedy approach is able to pair more critical points than a persistence-based simplification. In the above example (cf. Figure 4.4) all critical points could be paired with the greedy approach, and a sole minimum remains in V_m . The greedy approach leads to longer separation lines and larger separation surfaces in the coarsest level of the hierarchy. The coarsest level using the persistence pairing, on the other hand, may contain a large number of unimportant critical points.

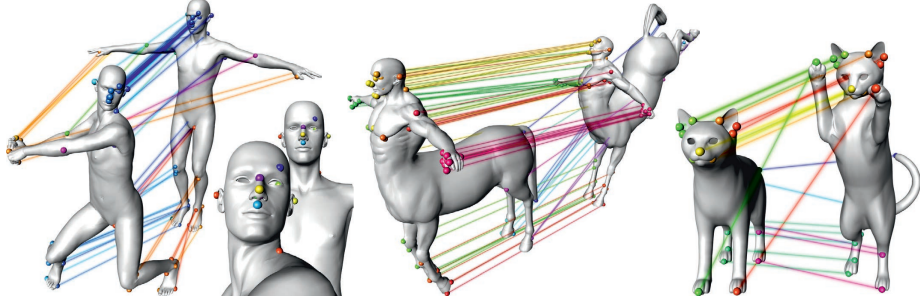


Figure 4.5: Matching results for different models, where correspondences are shown in the same color and connected by a line.

4.5 Application: Feature Point Correspondences

In the following, we present an application of persistent homology.

We can use persistence to find correspondences of persistent feature points on near-isometric surfaces. To effectively match feature points, they should not only be distinctive, but also intuitive and visually meaningful. This is important in visually evaluating correspondence quality on real data, where no ground truth correspondence is available for numerical evaluation, because dense matches derived from reliable sparse matches will degrade in accuracy as their distance increases from the set of reliable matches. Features should be derived from local surface properties, to allow for partial matches and incomplete surfaces. They should capture surface properties that depend only on the intrinsic geometry of the surface.

We can get a set of points with these properties from Gaussian curvature [Gau28]. For isometric surfaces, the Gaussian curvatures are identical. Therefore, the minima and maxima of Gaussian curvature fields are a good candidate set for feature points. However, Gaussian curvature is greatly affected by noise. This results in minima and maxima of which only a small subset describe meaningful features. A robust way to separate spurious minima and maxima from the important ones is by means of persistent homology. We leverage this power of persistence to extract reliable feature sets. In contrast to recently proposed techniques [SOG09], our features are efficient to compute and direct, not requiring a multi-resolution structure.

In the following, we present a pipeline for feature point correspondence between two surfaces and present straightforward applications of our pipeline to feature tracking in time-varying data and dense correspondence computation, thereby demonstrating stability and accuracy of our pipeline. We further demonstrate the effectiveness of our method by a thorough evaluation on synthetic data with noise, holes and major missing parts, and on real data. A gallery of some exemplarily results is given in Figure 4.5. The following section is based on the work [YGW⁺12].

4.5.1 Related Work on Shape Matching

There are many recent works on feature extraction and surface correspondence, so for conciseness we will only review the most relevant here. For a more exhaustive comparison of correspondence methods, we refer the reader to a recent survey [vKZHC011] and a recent competition [BBC⁺10].

Feature Extraction: Given a set of features on each surface, this reduces the search space for correspondences. Given feature correspondences, we can compute dense matches using seed growing [SHCB11] or front propagation [TM10].

The Heat Kernel Signature (HKS) [SOG09] organizes information about the intrinsic geometry of a shape in a multi-scale way that is stable under perturbations of the shape. Hence, it is an effective feature detector and descriptor. Features are detected as local maxima of the HKS for large scales. A later variant used persistent homology to filter out unstable feature points [DLL⁺10]. While we also use persistence to filter feature points, we apply discrete Morse theory to Gaussian curvature, thus making our approach less computationally costly and conceptually simpler. We demonstrate that the power of persistence to distinguish between features and noise allows us to use a simple isometry-invariant scalar field to extract features for correspondence.

The difference of Gaussians (DoG) and histogram of oriented gradients (HOG) feature operators have been adapted to meshes, MeshDoG and MeshHOG [ZBVH09], and applied to mesh matching. This method requires a multi-scale neighborhood structure, whereas we only need a fixed neighborhood to compute Gaussian curvature.

Correspondence: Möbius voting [LF09] uses the observation that isometries are a subset of the Möbius group to devise a method for automatic sparse surface correspondence. After mid-edge flattening to the complex plane, triplets of points are chosen from both point sets and a Möbius transform is computed in closed form. This is followed by a voting scheme that is weighted by the estimated deviation from isometry. A higher-order Markov random field (MRF) formulation of graph matching combined with a voting and clustering scheme based on Möbius transforms has also been proposed for both sparse and dense correspondence [ZWW⁺10]. This method handles large deformations, partial matching and changes in scale. Blended intrinsic maps [KLF11] find per-point blending weights for multiple low-dimensional intrinsic maps computed using Möbius voting; these maps are then blended by linear interpolation. This allows to search relatively small sets of possible deformations, yet still handle large, anisometric deformations. Accurate correspondence results compared to other recent methods are demonstrated for clean, genus zero meshes. We use these methods for comparison because they are the current state-of-the-art and have demonstrated equal or superior performance to competing algorithms [KLF11].

A feature-based dense correspondence method [TM10] starts by computing sparse feature correspondences, and then uses a MRF and front-propagation to compute dense correspondence. It is a well-established technique to extract feature points and explore permutations of matches to find a combination with minimal alignment and deformation error [HAWG08, ZSCO⁺08]. Two of these methods [TM10, ZSCO⁺08] use the geodesic integral or average to extract features, which are more expensive than Gaussian curvature. The work of Huang et al. [HAWG08] uses principal curvature, which is not isometry invariant.

Many methods make use of the isometry assumption, often using some kind of embedding [vKZHCO11], which are often indirect and expensive to compute. If the embedding is global then the method can be expected to have difficulty with partial matching. A method leveraging the isometric assumption to register partially corresponding surfaces that is robust to topological noise [TBW⁺09] samples the space of feasible feature matches to explicitly examine alternative solutions. This method and its extensions require significant preprocessing, whereas our method requires none.

Other methods based on the isometry assumption consider the heat kernel as an isometry-invariant local surface descriptor. A topologically robust method for dense

correspondence [SHCB11], that starts with a set of sparse correspondences as input, successfully transfers a scalar function between surfaces undergoing major topology changes. A heat kernel is computed at all points on both surfaces, and dense correspondences are computed by seed-growing from the sparse correspondences. The seed-growing algorithm is similar to the one we use in Section 4.5.3. Given one pair of corresponding points, full correspondence can be computed for two isometric surfaces using the heat kernel [OMMG10]. For every point on S , its image on \tilde{S} is characterized by the preservation of the heat kernel to the given corresponding points. Our features are derived directly from the surface and our descriptors are based only on geodesics, and therefore are not as computationally expensive as the heat kernel.

4.5.2 The Matching Pipeline

This section presents a pipeline to find feature point correspondences on two near isometric surfaces S and \tilde{S} . We call two surfaces nearly isometric if the ratio of any corresponding geodesic distances is bounded by a constant threshold τ . The idea of this pipeline is to extract features as the most dominant extrema of the Gaussian curvature in terms of persistence and to find near isometric correspondences between these feature sets using modifications of established algorithms.

Feature Points and Persistence

In this work, we interpret feature points as extremal points of a curvature field. Since we assume isometry, we use Gaussian curvature. In recent years, several techniques to compute this quantity were proposed. We use a simple quadratic least-square fitting to the underlying point cloud [CP03] to compute the Gaussian curvature. However, our pipeline does not depend on this choice.

We consider the scalar field formed by the Gaussian curvature on a surface. Points of minimal and maximal Gaussian curvature are critical points of this scalar field. A robust and consistent way to compute critical points is by means of discrete Morse theory. We use the Algorithm 5 to compute the critical points in a combinatorial fashion.

Numerical issues in the curvature computation and noise may create spurious critical points, which challenge the upcoming matching. To distinguish noise-induced and dominant critical points, we make use of the importance measure for critical points as introduced in Section 4.1: *persistence*. We denote the most dominant minima and maxima of the Gaussian curvature fields on the surfaces S and \tilde{S} as feature points F and \tilde{F} , respectively.

Computing Correspondences of Feature Points

Correspondences between feature points are found as follows. For each feature point, we construct a vector based on the geodesic distances between it and a set of sample points on the surface. We measure the similarity of these vectors and find initial correspondences by solving a minimization problem. We then enforce isometric consistency of the set of correspondence pairs using graph matching, pruning inconsistent matches. Finally, a post-matching method finds additional matches that are consistent with the established correspondences.

Initial Correspondences: We first find an initial correspondence between the feature sets by matching the spatial distribution of feature points. Looking from one feature

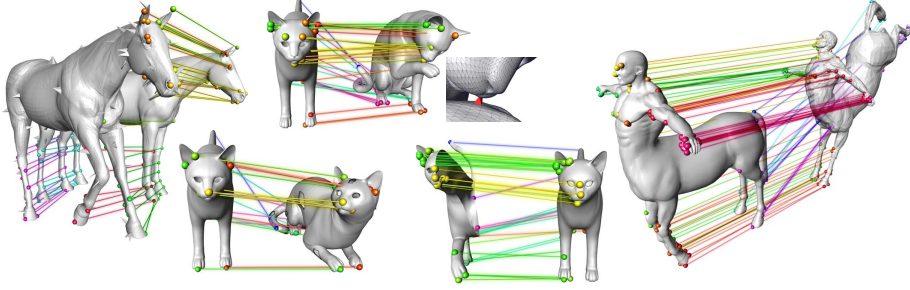


Figure 4.6: Matching results for different models corrupted by synthetic noise. From left to right: outliers, holes, topological noise, partial information, and Gaussian noise.

point x on S to a uniquely defined set of reference points Y , the distribution of those points depends on the point of view of x and is unique up to intrinsic symmetry. When measuring the distribution with an isometric quantity such as the geodesic distance, this point of view is invariant under isometric transformations. We represent the view point dependent distribution of Y in a quantitative manner by constructing two sets of reference points Y and \tilde{Y} of cardinality R from S and \tilde{S} using *geodesic farthest point sampling* [BBK08] and by considering the quantity $f(x, y) = 1/(1 + g(x, y))$, where $x \in F$, $y \in Y$, and $g(x, y)$ denotes the geodesic distance between x and y on S . The function f measures the influence of the reference points on each feature point, and is designed to allow for partial matching as nearby points are weighed more than distant points, and the local neighborhood therefore has a greater influence.

The *feature vector* \mathbf{f}_Y for a given feature point $x \in F$ is given by the collection of $f(x, y_j)$ for all reference points $y_j \in Y$ in non-decreasing order. Consider two surfaces S and \tilde{S} and their respective feature points F and \tilde{F} . Assuming $x \in F$ is the correspondence of $\tilde{x} \in \tilde{F}$, the corresponding feature vectors $\mathbf{f}_Y(x)$ and $\mathbf{f}_{\tilde{Y}}(\tilde{x})$ are expected to be similar. Hence, we measure the dissimilarity Ψ of two feature vectors by their normalized L^1 -distance. Computing the dissimilarity between all feature vectors of F and \tilde{F} yields a dissimilarity matrix. A good correspondence is found if the sum of all its dissimilarities is small. Therefore, the aim is to find a minimum assignment by column and/or row permutation to minimize the trace of the dissimilarity matrix. To solve this optimization problem, we use the Hungarian algorithm [Kuh55], which results in a set of correspondences Σ_1 .

Isometric Correspondences: In the following, we remove the pairs in Σ_1 that are not consistent with the assumption that deformations should be approximately isometric. We aim to find the largest set Σ_2 of consistent correspondences. These correspondences can be found using a kernel extraction method as proposed by Leordeanu and Hebert [LH05] and used by Huang et al. [HAWG08]. Let (c_i, \tilde{c}_i) denote the i -th correspondence in Σ_1 . Any two consistent correspondences $\{c_i, \tilde{c}_i\}$ and $\{c_j, \tilde{c}_j\}$ should satisfy the following near-isometry constraint: the minimum c_{ij} of the two ratios $g(c_i, c_j)/g(\tilde{c}_i, \tilde{c}_j)$ and $g(\tilde{c}_i, \tilde{c}_j)/g(c_i, c_j)$ should be larger than the stretching tolerance τ with $0 < \tau < 1$. It is known that a set of correspondences satisfying this condition can be found using a spectral method on a matrix M that depends on c_{ij} and τ . For more details, refer to [HAWG08].

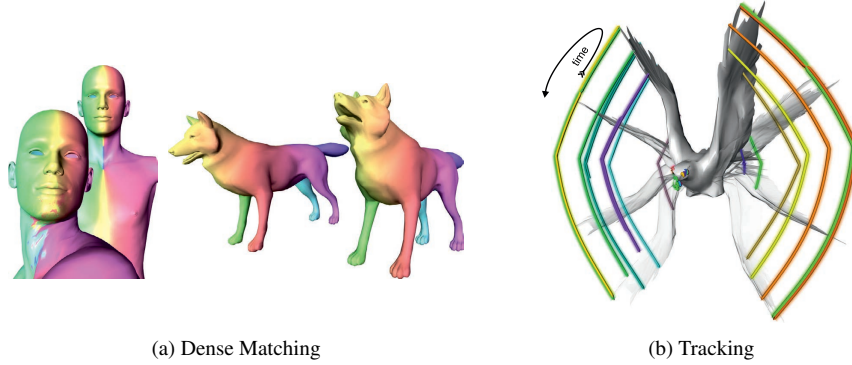


Figure 4.7: (a) Visualization of dense correspondence results. The same color is mapped from one surface to another using the dense correspondences. (b) Tracking of the *Eagle* sequence. Tracked features are shown as colored curves.

Final Correspondences: As the set Σ_2 might not contain all near-isometric feature point matches, we add additional pairs of feature points in the final step. The additional correspondences are found based on a modified geodesic triangulation technique. Let $F_R \subset F$ and $\tilde{F}_R \subset \tilde{F}$ denote the sets of the "rejected" feature points for which correspondences have not been found yet. For each point in F_R , we compute a feature vector w.r.t. the matched points similar to above. The only difference is that the feature vector is now ordered w.r.t. an arbitrary but fixed order of the correspondences in Σ_2 . We add a new correspondence pair if the feature vectors of two points are symmetric nearest neighbors under the dissimilarity measure Ψ and the new pair respects the isometric threshold τ w.r.t. all correspondences in Σ_2 . All pairs that fulfill these conditions are added to the set Σ_3 of feature correspondences, which is initialized by Σ_2 .

4.5.3 Applications

Dense Matching: The first application is dense matching, which considers all points except the correspondences as candidates and attempts to find correspondences for them. The dense matching approach performs seed growing on the set Σ_3 of coarse correspondences. It is a variant of the approach by Sharma et al. [SHCB11], and it is also closely related to our post-matching algorithm. Let $F_3 \subset S$ and $\tilde{F}_3 \subset \tilde{S}$ be the sparse corresponding point sets such that for every $c \in F_3$ there exists exactly one $\tilde{c} \in \tilde{F}_3$ with $(c, \tilde{c}) \in \Sigma_3$ and vice-versa. Let $\mathcal{N}_k(x)$ denote the set of k -ring neighbors of a point x on S . Consider z as an unmatched point on S , $(c, \tilde{c}) \in \Sigma_3$ and $z \in \mathcal{N}_k(c)$. We find a correspondence for z in the set of unmatched neighbors $\mathcal{N}_k(\tilde{c})$ of \tilde{c} using feature force vectors $\mathbf{f}_{F_3}(z)$. More precisely, we compare the feature force vectors $\mathbf{f}_{F_3}(z)$ and $\mathbf{f}_{\tilde{F}_3}(\tilde{z})$ for all $\tilde{z} \in \mathcal{N}_k(\tilde{c})$, and match z to the point in $\mathcal{N}_k(\tilde{c})$ that has the most similar feature force vector.

We visualize two dense matching results in Figure 4.7a. The same color is mapped from one surface to another using the dense correspondences. Note that a globally coherent correspondence is found even though some local mismatches occur, as can be seen at the neck of the human model.

Tracking: The second application is tracking, where we are given a sequence of

Table 4.2: Parameter settings for all models used in the tests.

	<i>Cat</i>	<i>Cat (topo. noise)</i>	<i>Centaur</i>	<i>David</i>	<i>Dog</i>	<i>Horse</i>	<i>Wolf</i>	<i>Face 1</i>	<i>Face 2</i>	<i>Face 3</i>
κ	0.05	0.008	0.05	0.05	0.04	0.03	0.05	0.04	0.10	0.02
τ	0.72	0.72	0.83	0.83	0.72	0.72	0.72	0.72	0.72	0.72

frames, extract feature points from these frames and find correspondences between the consecutive frames. This application requires not only stable detection of feature points, but also stable matchings between any two frames. Our goal here is not to define a new framework for tracking, but to demonstrate the stability of our feature correspondence, and hence we use a straightforward and direct use of our method for matching two surfaces. This has the additional advantage of not requiring to process the entire sequence as a batch, but rather allows the possibility to process it in sequence.

For demonstration, we use a time-varying synthetic model of an *Eagle* consisting of 100 frames from Martinez Esturo et al. [MERT12]. Fig. 4.7b visualizes the tracking results, where the tracked features are shown as colored curves.

4.5.4 Experiments

This section validates the proposed pipeline. We implement the pipeline using MATLAB and C++ and test it on a standard PC. We use code from Surazhsky et al. [SSK⁺05] to compute geodesics and code from Cao² for the Hungarian algorithm. Our non-optimized implementation takes about 3 minutes to find corresponding points for the *Cat* model and about 1.5 minutes for the *Centaur* model.

We evaluate the algorithm on a large number of models of the TOSCA [BBK08] database and some models of the BU-3DFE [YWS⁺06] database. Similar to Bronstein et al. [BBC⁺10], we define the correspondence error \mathcal{C} as follows:

$$\mathcal{C} = \frac{1}{|\Sigma_3| \cdot d_g} \min \left\{ \sum_{i=1}^{|\Sigma_3|} g(c_i, c'_i), \sum_{i=1}^{|\Sigma_3|} g(c_i, c''_i) \right\}, \quad (4.4)$$

where $|\Sigma_3|$ is the cardinality of Σ_3 , d_g is the geodesic diameter of neutral pose S , (c_i, \tilde{c}_i) is a correspondence pair in Σ_3 , c'_i and c''_i are the ground truth correspondence and the symmetric ground truth correspondence of c_i in S , respectively, and the geodesic distances $g(c_i, c'_i)$ and $g(c_i, c''_i)$ are measured on S . Here, the symmetric ground truth c''_i is defined as the ground truth mapping of c_i to its intrinsically symmetric part on the shape, given by flipping the left and right sides of the model.

Our algorithm involves three parameters: the persistence threshold κ , the cardinality R of the set Y of sample points and the deformation threshold τ . Regarding parameter settings, we fix $R = 800$ and each of the other two parameters at one consistent value per model, with the exception of the topological noise example, as shown in Table 4.2.

Figures 4.9a and 4.9b shows the influence of the κ and τ parameter values on the results of matching two *Cat* models. A correspondence c_i is considered close to its ground truth c'_i if $g(c_i, c'_i) < 0.05 d_g$, close to its symmetric ground truth c''_i if $g(c_i, c''_i) < 0.05 d_g$, and a mismatch otherwise. As expected, as the persistence threshold κ increases, the number of features decreases and as the stretching threshold τ increases, the number of matched pairs decreases.

²<http://www.mathworks.com/matlabcentral/fileexchange/20328>, 2008

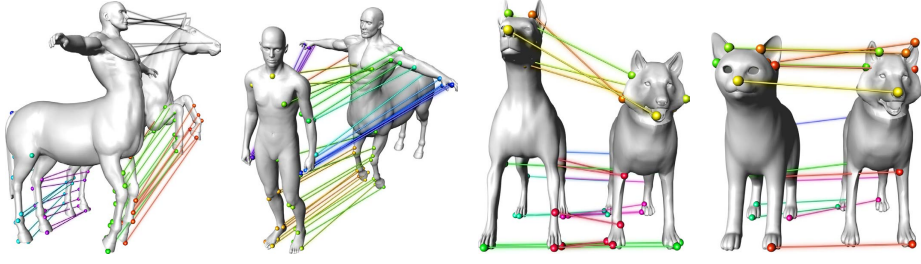


Figure 4.8: Matching between different object classes.

Synthetic Evaluation

We evaluate the robustness of our algorithm on TOSCA models from four perspectives: isometric deformation, different categories of noise, different object matching and partial matching. Whenever we match two shapes from the same object class, we match the deformed/noisy model to the clean shape of the same object class in neutral pose.

To evaluate the robustness against noise, we artificially introduced five different kinds of noise to some models: Firstly, we added three levels of *Gaussian noise* to the deformed versions of the *Cat*, *Centaur*, *David*, *Dog*, *Horse*, and *Wolf* models (38 models total). The variances of Gaussian noise used in the experiments are 20%, 40% and 60% of the model's bounding ball radius.

Secondly, we added three levels of *outliers* to the aforementioned 38 models by moving a vertex in the direction of its outer normal with probability 0.004 by varying the strength of the offset. The outliers are modeled as a type of shot noise that is typically present in scanner data from multi-view camera systems. The models are corrupted by moving a vertex in the direction of its outer normal with probability 0.004. We use three levels of outliers by varying the strength of the offset.

Thirdly, we added three levels of *holes* to the deformed versions of the *Cat* model (10 models total). The first level removes the one-ring neighborhood of a set of vertices distributed over the surface. The second and third levels enlarge the holes by removing all triangles that are on the boundary of the model.

Fourthly, we removed parts of the models in three levels to simulate *partial matching*. For each model in neutral pose, we removed a part by cutting the model with a plane parallel to the symmetry plane of the model. The three levels represent the removal of 17%, 33%, and 50% of the model's bounding box, respectively. The partial models are then deformed into all other poses to generate all partial models.

Finally, we added *topological noise* to one of the *Cat* models.

Figure 4.9c shows the correspondence errors \mathcal{C} for the *Cat* models with near-isometric deformations and different types of noise. Note that the correspondence quality does not degrade significantly for increasing levels of Gaussian noise, outliers or holes. As expected, for increasing levels of partial matching, the quality of the correspondence degrades more than for the other types of noise. However, even in case where 50% of the surface was removed, the average correspondence error is below 30% of the geodesic diameter.

Non-Isometric Deformation: Figure 4.9d show \mathcal{C} for the correspondences computed between pairs of *Cat* models. For all of the models that have a mean correspondence error above 0.03, we encounter the following problem. Some points on S correspond to points close to their ground truth correspondences on \tilde{S} , while other points on S

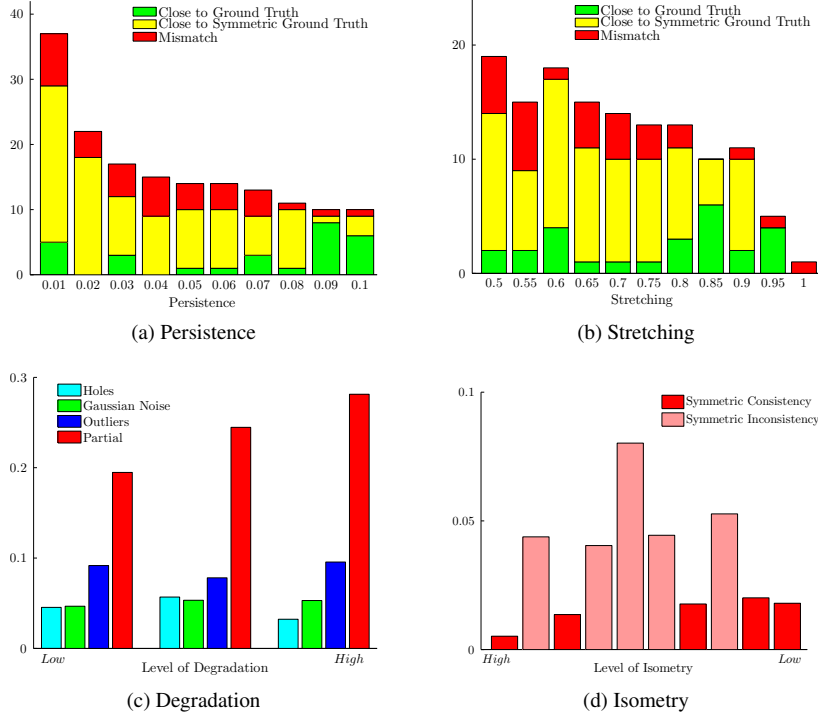


Figure 4.9: (a,b): Influence of parameter values on matching two clean *Cat* models. The x -axes show the thresholds and the y -axes show the number of matches. (c,d): Correspondence error \mathcal{C} for the *Cat* model with different types and levels of degradation. Each bar in (c) is the mean over all model pairs, while each bar in (d) is for one pair.

correspond to their symmetric ground truth correspondences on \tilde{S} . Hence, while all correspondences are locally acceptable, the correspondence map is globally inconsistent, which leads to a large value of \mathcal{C} . We call this problem *symmetric inconsistency* in the following, and the matching of the legs of the *Dog* and *Wolf* models in Figure 4.8 shows an example. However, the symmetric inconsistency only affects very few feature points as can be seen in the bar plots of Figures 4.9a and 4.9b. The majority of feature points are correctly matched w.r.t. the ground truth or the symmetric ground truth. Figure 4.5 shows some qualitative results.

Gaussian Noise: As Gaussian noise will change the intrinsic geometry of the shape, we adjust the parameter τ depending on the specific level of noise. Basically, τ decreases with the increase of the noise level. This is because stronger noise will create a greater deformation than weaker noise does. Figure 4.6 illustrates matching a *Centaur* model degraded by Gaussian noise and its corresponding clean model in neutral pose. As demonstrated by both the small correspondence errors in Figure 4.9c and the qualitative results in Figure 4.6, our method is able to match feature points even in the presence of Gaussian noise.

Outliers: Figure 4.6 shows an example of matching a *Horse* degraded by outliers and its corresponding clean model in neutral pose. Both the numerical evaluation in

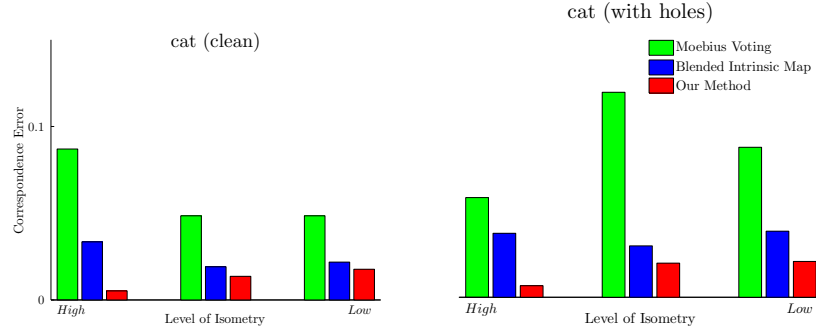


Figure 4.10: Comparison for *Cat* without noise and with second level of holes.

Figure 4.9c and the qualitative results in Figure 4.6 demonstrate that our algorithm is able to find high-quality correspondences, when applied to data with outliers.

Holes: Figure 4.9c shows that our method still provides comparable performance as for clean models in terms of correspondence quality, although the existence of holes might potentially result in significant changes in geodesic paths. Figure 4.6 illustrates qualitatively that feature points are correctly matched.

Partial Matching: Figure 4.6 shows an example of matching a partial *Cat* model to a complete *Cat* model. The feature points appearing in both models are visually matched correctly. In a second experiment, we are interested in finding corresponding pairs of vertices for shapes from different object classes, where parts of the shapes are near-isometric and other parts are not. The partial matching results are illustrated in Figure 4.8 (left). We observe that feature points describing semantically the same region are correctly matched. For instance, observe that the hands and upper body between *Centaur* and *David*. However, feature points could not be matched correctly in regions of the surfaces that are semantically different, as expected. This can be seen in Figure 4.8 at the head of *Horse* and the head of *Centaur*.

Topological Noise: Topological noise significantly changes the intrinsic geometry of the surface, and is thus expected to cause problems for our algorithm. Figure 4.6 shows the correspondences on a *Cat* model with topological noise.

Different Object Matching: Finding correspondences between two objects of different classes is challenging, since the surfaces are far from isometric. However, our pipeline is able to match most of the feature points correctly, as can be seen in Figure 4.8 (right).

Comparison to Prior Methods

We compare the proposed method with two state-of-the-art matching algorithms: Möbius voting [LF09] and blended intrinsic maps [KLF11]. To this end, we applied the two methods to pairs of models of the same object from the TOSCA dataset. For the comparison (Figure 4.10), we only use pairs of models for which our method does not encounter the symmetric inconsistency problem. To compute the results for Möbius voting and blended intrinsic maps, we use the code released by the authors [Kim]. Note that the implementation of Möbius voting may not reflect all the details of the original implementation. Figure 4.10 shows the correspondence errors \mathcal{E} for models

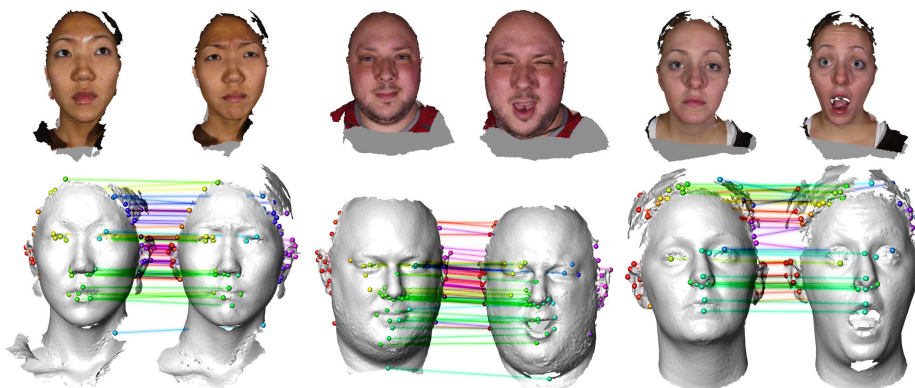


Figure 4.11: Correspondences of scans with different facial expressions. The top row shows the textured raw scans and the bottom row shows our results.

with non-isometric deformations and with holes. Note that our method generally compares favorably to previous approaches. We observed this trend for different models and types of noise in our experiments.

Real-world applicability

To assess the real-world applicability of our algorithm, we compute correspondences between different face scans from the BU-3DFE database. These tests are challenging because the meshes have inconsistent topology and different local shape features. The results here are presented visually, as no ground truth is available to evaluate numerically. The matching results are shown in Figure 4.11. Note that accurate correspondences are found for different face shapes and expressions.

4.5.5 Discussion

We presented an algorithmic pipeline for finding correspondences of persistent feature points on near-isometric surfaces. As shown in Section 4.5, the pipeline combines many useful properties:

- It is built from established components.
- It produces accurate and stable correspondences.
- It is conceptually direct and simple.
- It is computationally efficient.

There are some limitations of our approach:

- The symmetric inconsistency problem shown in Figure 4.8 (right). This problem is caused by the construction of the descriptors for feature points based on geodesic lengths whose influence is inversely proportional to the distance. This means that points that are far from a feature have little influence on its descriptor. While this aids our method in computing correspondence information between partially near-isometric surfaces, it causes this limitation.

- The pipeline cannot stably compute high-quality correspondences when applied to models with topological noise, see Figure 4.6. Topological noise significantly changes the intrinsic geometry of a surface. Since our algorithm heavily uses the assumption of isometry-invariance, these changes cause problems. To overcome this, we include more feature points in this example by changing the parameter value of κ to 0.008.
- The proposed pipeline may not find satisfactory correspondences for all features in case of non-isometric shape matching, see Figure 4.8, because the two shapes could have significantly different intrinsic geometry.

Despite these drawbacks, we believe that our pipeline allows for a straight-forward computation of correspondences between persistent feature points. We show that our method is robust against isometric deformations and different types of noise, including Gaussian noise, outliers, holes, topological noise and scanner noise. Moreover, satisfactory correspondences can be found even in the case of partial matching. We leave it for future work to devise strategies to overcome the symmetric inconsistency problem encountered when matching surfaces that have symmetric structures.

Chapter 5

Quantification of Separatrices

In Chapter 3, we introduced a strategy to create different levels of details of the Morse-Smale complex. This strategy allowed us to reduce the Morse-Smale complex to its essential topological structures representing the large-scale behavior of the data.

However, the simplification process of the Morse-Smale complex involves binary decisions. A separatrix is either completely removed from the complex or not. In many application cases, this simplification is only a rough indicator whether a separatrix is important or not. Parts of a separatrix might be unimportant but others may be essential. From the application point of view, an important image-based feature might be missed due to the removal of the entire separatrix. Therefore, it is necessary to assess each point of a separatrix individually by an importance measure.

Persistent homology as introduced in Chapter 4 assigns an importance value to each critical point of the input data. Based on this importance, spurious critical points can be distinguished from the dominant ones. An important justification of persistence is the stability result of Cohen et al. [CSEH07]. This result states that persistent homology is stable with respect to noise.

In this chapter, we introduce the concept of *separatrix persistence*. This is an importance measure for every point along a separation line and surface, which allows us to tell apart their dominant parts from spurious ones. The technique is founded on the hierarchy of the Morse-Smale complex and persistent homology. It benefits from the global nature of these two concepts and their stability against noise. The information provided by persistence is propagated along the higher dimensional topological structures. Due to the simplification process, the propagation takes the large-scale behavior of the features into account.

As discussed in Chapter 1, two different concepts are commonly used for extracting extremal structures: the local analysis due to ridges and valleys, and the global point of view by means of topology. In this chapter, we concentrate on the topological view. The extremal structures are herein covered by the topological structures and form a subset of them. We compare our extraction results to the results of a local analysis. Therefore, we provide a brief discussion of the similarities and differences of the topological structures to ridges and valleys in Section 5.1. We will see that both concepts extract features with an extremal characteristic. However, these characteristics differ and depend on the point of view.

After this discussion, we introduce separatrix persistence in Section 5.2. We first define this new importance measure for the 2-dimensional case. In this case, we benefit from an optimal topological simplification based on the Morse inequalities (Sec-

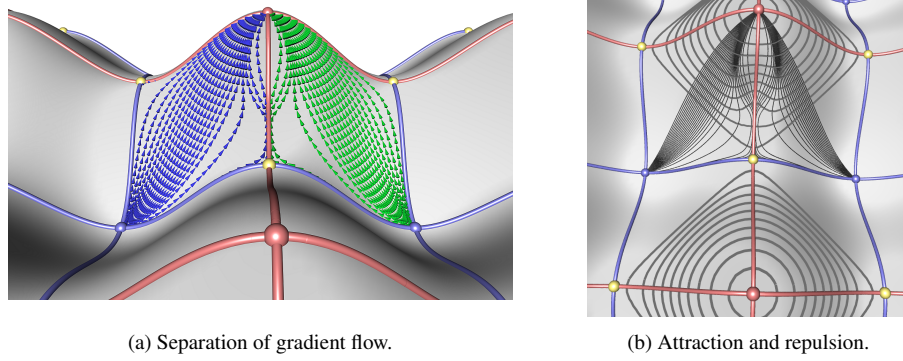


Figure 5.1: Illustration of the analytic function f (5.1): The minima, saddles and maxima are depicted as blue, yellow, and red spheres. The 0- and 1-separation lines are shown as blue and red lines, respectively. The integral lines of the gradient ∇f are illustrated by green and blue arrow lines in (a). The local attracting and repelling behavior of a separatrix can change as illustrated by the central separatrix in (b). The bending of the gray isolines in (b) depicts a change from a local convex to a local concave behavior of f . However, the gradient flow is still separated by the separatrix (central red line).

tion 2.2). This will help us to convey the idea of the proposed technique. We then extend this theory to three dimensions. As discussed in Section 4.4, the pairs of critical points defining the hierarchy do not need to coincide with the persistence pairs. This needs to be respected by the extension of separatrix persistence to three dimensions.

Separatrix persistence allows us to extract the extremal structures described by the Morse-Smale complex: the most dominant (parts of) separation lines and separation surfaces. In Section 5.3, we apply this importance measure to a wide range of applications thereby illustrating its robustness and applicability. We want to emphasize that such structures are not isosurfaces or isolines of the examined scalar field, since the scalar value varies on them.

The concept of separatrix persistence and its applications is based on the published works [WG09, GMW⁺10, GSW12, PGW12].

5.1 Separatrices and Height Ridges

Two different concepts are commonly used for extracting extremal structures: the local analysis due to ridges/valleys and the global point of view by means of topology. In this work, we concentrate on the topological view. Nevertheless, we provide a discussion of the similarities and differences between the local and the global approach in the following.

A minimal/maximal *point* is canonically defined in arbitrary dimensions. However, its higher dimensional generalizations cannot be defined in a canonical way, i.e., several equated definitions exist for extremal *lines* or *surfaces*. This is documented throughout the literature [KvD93, Dam99, LLSV99, SWTH07, PS08, SPFT12]. Besides their global definition as topological separatrices [Max70], another frequently used concept are *Ridges/Valleys*, which goes back to De Saint-Venant [dSV52]. The

relationship between these mathematically different approaches (local vs. global definition) was debated vigorously in the computer vision community in the early 1990s (see, e.g., Koenderink and van Doorn [KvD93]). Nowadays, there is a consensus that both approaches have their merits, and López et al. [LLSV99] provide an exhaustive evaluation of the equated local and global definitions.

A recent variant of ridges/valleys is the *Height Ridge* definition [Ebe96]. This definition is local and builds on the first and second derivatives of f , i.e., the gradient ∇f and the Hessian Hf . The local ridge definition investigates the convexity/concavity of f in a local neighborhood of a given point. As elegantly formulated by Peikert and Sadlo [PS08], ridge lines in a scalar field are found at locations where the vectors ∇f and $Hf \cdot \nabla f$ are parallel. They can be extracted using the Parallel Vectors operator [PR99, POS⁺11]. In case of 3-dimensional scalar fields, ridge surfaces can be found as parts of the zero level set $\nabla f \cdot \mathbf{e}_1$ with $\lambda_1 < 0$, where \mathbf{e}_1 is the eigenvector to the smallest eigenvalue λ_1 of Hf . A consistent orientation of the eigenvectors at the vertices of each cell is necessary to extract this level set. This can be achieved using a principal component analysis [FP01].

In contrast, separatrices represent the border of adjacent compartments where f behaves monotonically [Sma61b]. The separatrices literally separate the gradient flow within these compartments. Let us consider the following example: the analytic function $f : [-1, 1]^2 \rightarrow \mathbb{R}$ consisting of an anisotropic Gauss function g , a local distortion d , and a rotation r with angle $\theta = \pi/4$:

$$\begin{aligned} g(x, y) &= 0.05 e^{-(200x^2 + 300(y - 0.1768)^2)} \\ d(x, y) &= x + g(x, y) (\cos(5x) \sin(20y + 10.6066)) \\ r(x, y) &= \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \begin{pmatrix} d(x, y) \\ y \end{pmatrix} \\ f(x, y) &= \sin(2\pi r_1(x, y)) \cos(2\pi r_2(x, y)) \end{aligned} \quad (5.1)$$

Figure 5.1 provides an illustration of this example. Consider f as a height field and assume that a separatrix is given connecting a saddle with a maximum, see Figure 5.1a. Clearly, the central separatrix separates the flow (indicated by the arrows) of the left and right side. Imagine that rain pours onto this terrain and water assembles around the two minima. The water is rising and the shape of the water level is defined by the integral lines of the flow. If the water level continues to rise, it will reach the saddle at some time. This is the lowest point of separation. Continuing the rainfall, the two water basins meet themselves along the separatrix. The water on the left side is separated from the water on the right side. The coloring of the arrows in Figure 5.1a indicates this. This separation represents a (weak) monotony break of f .

The monotony break that can be observed in Figure 5.1a is not locally given. The saddle point is connected to two maxima and to two minima. These two minima are essential in order to understand the extremal characteristic of the central separatrix. Each of the two minima gives rise to compartments that meet each other along the separatrix. In order to check for a monotony break, one needs to investigate if a set of particles inserted in the gradient flow assembles in the same minimum or not. In contrast to the local ridge definition, this kind of characteristic is global and cannot be determined locally.

In fact, this global nature of a separatrix makes it very useful for feature extraction. Real-world data sets are usually affected by noise. The small fluctuations caused by the noise create local distortions similar to the example shown in Figure 5.1b. Within

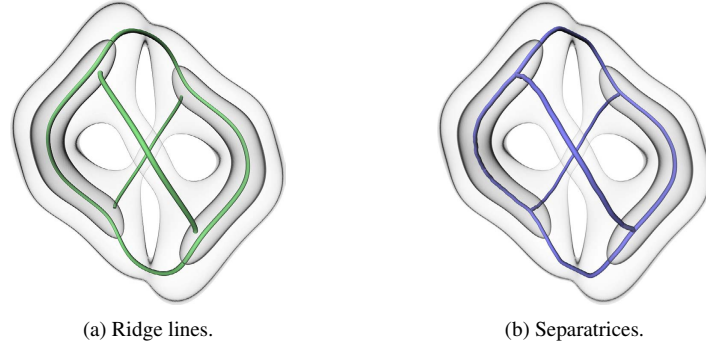


Figure 5.2: Extremal lines of the scalar field (3.1) following two different definitions.

small regions, the local attracting/repelling behavior of a separatrix can change. This happens when the local landscape described by f changes from convex to concave, or vice versa. However, its global nature prevents interruptions along the separatrix in contrast to the local ridge/valley definitions. In order to interrupt the course of a separatrix, the distortion must be so strong that new critical points are introduced.

It has been pointed out by Sahner et al. [SWTH07] that every separatrix can be assigned a ridge counterpart: each saddle point of f gives rise to ridges as well as separatrices (they do not need to coincide at any other places). However, not every ridge can be assigned a separatrix counterpart [SWTH07]. Intuitively, this happens when a ridge-creating fluctuation of f – as shown in Figure 5.1 – does not break its monotony. This is also nicely shown by the “Ridges without Critical Points” example of Peikert and Sadlo [PS08].

By definition (Section 2.2), separation lines are tangential to the gradient ∇f . Ridge lines, on the other hand, are defined as features where ∇f is parallel to $Hf \cdot \nabla f$. In fact, additionally requiring that they are also tangential to ∇f yields an overdetermined system as discussed by Schindler et al. [SPFT12].

There are several differences between ridges and separatrices from an algorithmic point of view: ridges are local features based on the first and second derivatives, which eases their extraction using parallel algorithms. This might be difficult for separatrices due to their global nature. On the other hand, separatrices can be extracted combinatorially without any derivatives.

Figure 5.2 shows the ridge lines (left) and separation lines (right) of the three dimensional scalar field defined by Equation (3.1) on page 53. It can be seen that they largely coincide and that both are in the center of the shown iso-surfaces which confirms their extremal characteristic. Interestingly, this example indicates that the local and the global approach besides their different definitions can give very similar results. Therefore, we will compare our topological approach in the subsequent sections to the local analysis of extremal features.

5.2 The Persistence of a Separatrix

In the following, we introduce the concept of the *persistence of a separatrix*. This concept is based on the multi-level representation of the Morse-Smale complex of a scalar function and its persistent homology. The separation lines and surfaces form

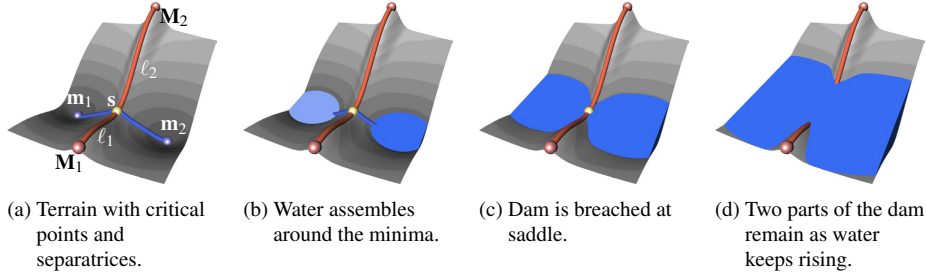


Figure 5.3: Separatrix persistence assigns an importance weight to each point along a separatrix. It respects that the importance of a feature may smoothly change along the line.

a superset of the desired extremal structures. Our new importance measure allows us to reduce the separatrices to this essential subset. This reduction does not require any derivatives.

We begin with the definition for the 2-dimensional case. In this case, we benefit from an optimal topological simplification based on the Morse inequalities (Section 2.2). Then, we extend the notion to three dimensions. Before we start, we introduce some notations used in the subsequent sections.

In the following, we assume that a Morse-Smale function $f : \Omega \rightarrow \mathbb{R}$ with $\Omega \subset \mathbb{R}^3$ is given [Sma61b]. The domain Ω is either given as a simplicial or cubical complex of dimension two or three. As described in Chapter 3, a multi-level representation of the Morse-Smale complex of f is given by a hierarchy of combinatorial gradient fields $\mathcal{V} = (V_i)$. To simplify notation, we define the *height difference* h of two points $x \in \mathbb{R}^3$ and $y \in \mathbb{R}^3$ as

$$h(x, y) = |f(x) - f(y)|. \quad (5.2)$$

To emphasize the relationship of the topological structures to the extremal structures, we sometimes call separation lines also *extremal lines*. Especially, lines that connect a saddle with a minimum are called *minimal lines*, and separation lines connecting saddles with maxima are called *maximal lines*. The persistence pairs and their importance is computed as described in Section 4.3. Given such a persistence pair (z, w) , we denote their persistence by $P(z)$ and $P(w)$. We want to stress that there holds: $P(z) = P(w) = h(z, w)$ for each persistence pair (z, w) .

5.2.1 The 2-dimensional Case

Previous schemes assigned only a constant value to the separation lines, i.e., the persistence of the saddle at which the separatrices emanate. To see why this does not suffice if we are primarily interested in extremal lines, consider the following intuitive example: Figure 5.3a shows the topological structures of a simple 2-dimensional scalar field on a terrain. Imagine that rain pours onto this terrain and consequently the water assembles around the two minima as shown in Figure 5.3b, where the blue surfaces denote the current water level. In this particular example, the water level represents the current persistence value during simplification. The two regions around the minima are still separated from each other by a “dam”, i.e., the separatrices ℓ_1, ℓ_2 (red). Water is rising and at some time it will reach the lowest point of the dam, i.e., the saddle. From

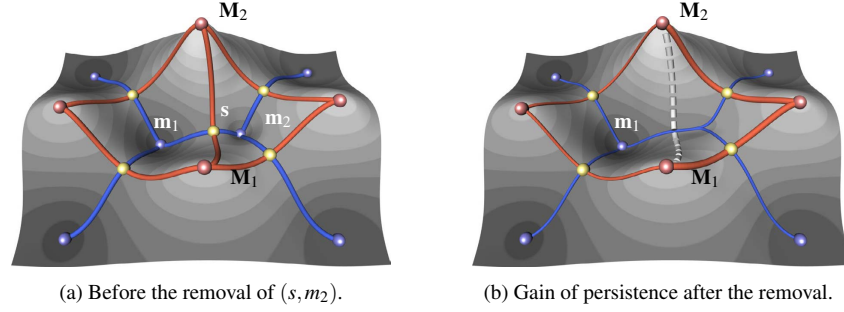


Figure 5.4: The cancellation of (s, m_2) assigns a final persistence to the removed separatrixes (gray, dashed). Neighboring separatrixes gain persistence (red, thick).

this moment on, water from both regions gets mixed, i.e., the regions are merged by canceling (s, m_2) and keeping m_1 (Figure 5.3c). Previous simplification schemes would also completely discard both red separatrixes ℓ_1, ℓ_2 at once. However, most of the dam is still there and remains to be there even if water keeps rising (Figure 5.3d). The two remaining parts of the dam will only be completely gone once the water level reaches their highest points, i.e., the maxima M_1, M_2 .

We draw two conclusions from these observations: Firstly, ℓ_1 and ℓ_2 have to be treated independently from each other after their saddle has been canceled. Secondly, the slow decay of each separatrix has to be described by an interval and not a single value. To do so, we give the following definition of the *local strength of a separatrix* I_{2D} :

$$I_{2D}(x) = h(x, s) + P(s). \quad (5.3)$$

The local strength I_{2D} measures the significance of every point on a separatrix for a given combinatorial gradient field. As it is derived from classic persistence, it inherits the stability under small perturbations. Obviously, I_{2D} reaches its highest value at the extremum. The point on ℓ with the lowest persistence is the saddle s since $h(s, s) = 0$. Note that the value of $I_{2D}(s)$ depends on whether ℓ is a maximal or minimal line. This is very important as it allows us to treat minimal and maximal lines independently: for example, an important maximal line can be “crossed” by a number of spurious minimal lines.

During topological simplification, the importance of affected separatrixes changes. Figure 5.4 illustrates the effect of the saddle-minimum cancellation (known from Figure 3.9) on the importance of the separatrixes in the neighborhood of the removed minimum m_2 : since the saddles on the right side of the domain have been reconnected to the stronger minimum m_1 , their red separatrixes *gain* persistence (shown as thick red lines). A similar statement holds for saddle-maximum cancellations and blue separatrixes. This change of importance is very essential since the repeated execution of simplification steps distills the large-scale behavior of f . Therefore, the overall strength of separation for a separatrix ℓ is given by taking the maximum of Equation 5.3 over all levels of detail $\mathcal{V} = (V_i)$:

Definition 1 (Separatrix Persistence 2D) *Given is a hierarchy of combinatorial gradient fields $\mathcal{V} = (V_i)_{i=0,\dots,m}$. Let ℓ be a separation line emanating at a saddle s at a hierarchy level i . The persistence of the separatrix ℓ is defined for each point $x \in \ell$ as*

$$S_{2D}(x) = \max_{i=0,\dots,m} I_{2D}(x). \quad (5.4)$$

As already described, the connectivity of the critical points changes after a simplification, see Figure 5.4. This also increases the length of adjacent separatrices (blue lines). A point x that lies on a separatrix is therefore connected to multiple saddles if we consider the complete sequence \mathcal{V} . Since the local strength I_{2D} is based on the persistence of the saddle for a single gradient field, the maximum in Equation 5.4 takes the most persistent saddle over the entire sequence \mathcal{V} into account.

The weight of the hierarchy \mathcal{V} is monotonically increasing in two dimensions (Section 3.5.5). Therefore, the persistence of a separatrix is given by its local strength at the moment when it is removed from the combinatorial gradient field. We combine the computation of separatrix persistence with the process of topological simplification in order to build up a feature hierarchy, i.e., determine the most important extremal lines. To do so, we simplify the combinatorial gradient field as described in Section 3.5.1 and compute S_{2D} when separatrices are removed from V (shown as gray dashed lines in Figure 5.4).

The computation of the persistence values of all separatrices is a global process since we coupled it to topological simplification. But not only from an algorithmic point of view: note that in general the distances between connected critical points increases during simplification. The last elements of \mathcal{V} contain those topological structures that represent the large-scale behavior of f .

5.2.2 The 3-dimensional Case

We now extend the notion of separatrix persistence to three dimensions, which assigns an importance measure to each point of a separation line and surface.

We need to consider two differences compared to the 2-dimensional case: Firstly, an optimal simplification based on the Morse inequalities of a Morse-Smale complex cannot be expected anymore, see Section 3.5.5. Hence, the coarsest level of detail in \mathcal{V} probably contains insignificant and spurious topological structures. Secondly, the hierarchy is no longer based on persistent homology. As discussed in Section 4.4, a pair of critical points which is removed in the simplification process is not necessarily a persistence pair. The definition of separatrix persistence in three dimensions will address these facts.

Similar to the 2-dimensional case, we first assess the local strength of separation by considering the evolution of isocontours. Consider a single 1-saddle s and its combinatorial separation surface S as shown in Figure 5.5. The separation surface is the boundary between the two volumes governed by the minima m_1 and m_2 . Our goal is to define the strength of this separation for each point on S . To do so, we observe how the evolution of an isocontour affects the separation between the volumes. Let $f(m_1) > f(m_2)$. Also note that $f(s) > f(m_1)$ by construction. For an increasing iso-value r , we have the following behavior for the isocontour:

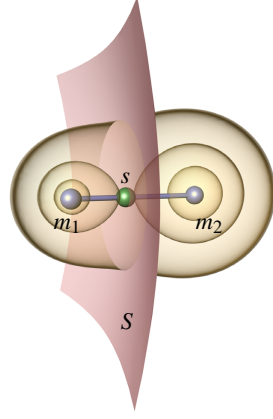


Figure 5.5: Illustration of the local feature strength of separation surfaces. The evolution of isocontours is depicted as yellow surfaces.

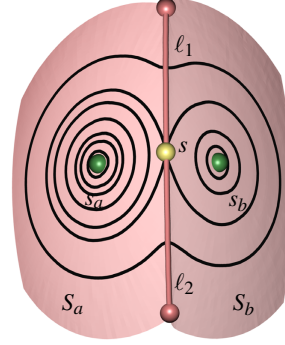


Figure 5.6: Illustration of the local feature strength of separation lines. The evolution of isolines is depicted as black lines.

zero components	$r < f(m_2) < f(m_1) < f(s)$
one component around m_2	$f(m_2) \leq r < f(m_1) < f(s)$
two components around m_1 and m_2	$f(m_2) < f(m_1) \leq r < f(s)$
the two components merge at the saddle s	$f(m_2) < f(m_1) < r = f(s)$
one component intersecting the separation surface	$f(m_2) < f(m_1) < f(s) < r$

The separation surface is pierced by the isocontour for the first time when the two components merge at the saddle. This infinitesimal small hole constitutes a breach of the separation between the two volumes. In other words, the saddle is the weakest point of separation between the two volumes. With further increasing r , the hole becomes larger, and we find that the outer parts of S provide the strongest separation between the volumes. Mathematically speaking, we define the *local strength of separation* I_{3D_S} for all points $x \in S$ as

$$I_{3D_S}(x) = P(s) + h(x, s), \quad (5.5)$$

which has its smallest value at the saddle: $I_{3D_S}(s) = P(s)$ denotes the persistence of s and thereby the “life time” of the weakest point on the separation surface. Note that $P(s) = P(m_1) = h(s, m_1)$, if (s, m_1) is a persistence pair, which is the case in a simple scalar field as described above. Later on, we will consider scalar fields with more topological structures, where (s, m_1) may not be a persistence pair. A statement similar to (5.5) holds for combinatorial separation surfaces emanating at 2-saddles.

The definition for the two separation lines ℓ_1, ℓ_2 of a saddle s follows the same ideas, except that these lines do not separate volumes, but areas on two neighboring separation surfaces S_a, S_b coming from two saddles s_a, s_b with $f(s_a) > f(s_b)$ (Figure 5.6). Therefore, we observe the evolution of isocontours of f restricted to these surfaces, i.e., we consider isolines. They emanate at s_a and s_b , merge at s , and create an

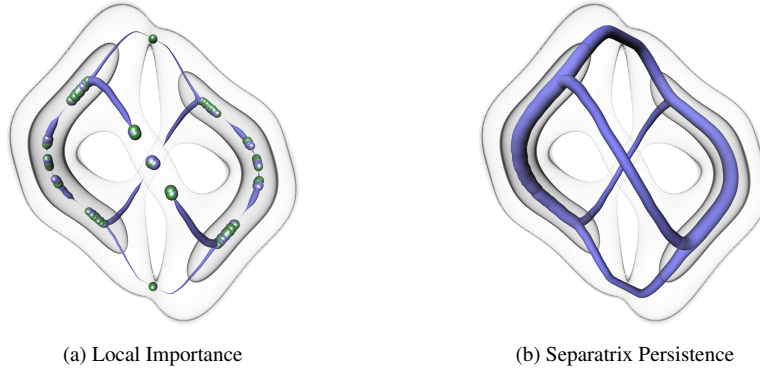


Figure 5.7: Extremal lines scaled by local feature strength (left) and separatrix persistence (right). Small fluctuations in the data cause an improper representation of the dominant extremal structures using the local feature strength. Separatrix persistence, in contrast, reveals the global structure.

increasingly larger hole in ℓ_1 and ℓ_2 . It turns out, we can define the *local strength of separation* I_{3D_ℓ} for all points $x \in \ell_1 \cup \ell_2$ very similar to (5.5):

$$I_{3D_\ell}(x) = P(s) + h(x, s). \quad (5.6)$$

The use of the local strengths of separation I_{3D_ℓ} and I_{3D_S} does not accommodate the global gestalt of the function; as shown in Figure 5.7a for a synthetic data set. Local perturbations cause an erratic and unintuitive behavior of I_{3D_ℓ} and I_{3D_S} – if applied directly to the unsimplified Morse-Smale complex.

We use the hierarchy of combinatorial gradient fields $\mathcal{V} = (V_i)$ from Section 3.5.3 to successively remove small perturbations and gain an increasingly global view of the topological features. Note that the connectivity of critical points changes within \mathcal{V} . Additionally, combinatorial separation lines and surfaces may merge during the simplification process. A point x on a separatrix can therefore separate multiple critical points. Hence, we need to determine the *maximal* strength of separation for x by considering (5.5) and (5.6) over all elements of (V_i) . We define for separation surfaces:

Definition 2 (Separatrix Persistence 3D for Surfaces) *Given is a hierarchy of combinatorial gradient fields $\mathcal{V} = (V_i)_{i=0, \dots, m}$. Let S be a separation surface. At a hierarchy level i , the surface S emanates at a saddle s . At most two extrema are connected to the saddle s at level i : let e denote the extremum with the smallest persistence. The Separatrix Persistence S_{3D} is defined for each point $x \in S$ as*

$$S_{3D}(x) = \max_{i=0, \dots, m} (P_{\max}(s, e) + h(x, s)), \quad (5.7)$$

where $P_{\max}(\cdot, \cdot)$ denotes the maximal persistence of two critical points.

In other words, $S_{3D}(x)$ is the *largest* strength of separation that could be found over

all hierarchy levels at the point x . This corresponds to (cf. Equation (5.5))

$$S_{3D}(x) = \max_i (I_{3D_S}(x)) \quad (5.8)$$

$$= \max_i (P(s) + h(x, s)) \quad (5.9)$$

$$= \max_i (P(e) + h(x, s)) \quad (5.10)$$

but only if the saddle-extremum pairs (s, e) obtained by the hierarchy are actually persistence pairs. As discussed in Section 4.4, this is not necessarily the case in 3-dimensional scalar fields. This may yield the situation that a saddle point with a low persistence is connected to an extremum with a high persistence. Taking only the persistence of the saddle into account would result in an underestimation of the emerging separatrices. We therefore use the maximum of persistence $P_{\max}(s, e)$ in (5.7) of two neighboring critical points (s, e) , since it estimates the largest strength of separation.

Note that a separatrix exists only up to a level V_j in the hierarchy (V_i) . Hence, (5.7) is effectively computed for the levels V_0, \dots, V_j , and not further examined for levels $k > j$.

Separatrix persistence for separation lines follows a similar scheme:

Definition 3 (Separatrix Persistence 3D for Lines) *Given is a hierarchy of combinatorial gradient fields $\mathcal{V} = (V_i)_{i=0, \dots, m}$. Let ℓ be a separation line. At a hierarchy level i , the separation line ℓ emanates from a saddle s . From the saddle s a separation surface emanates which has several other saddles in its boundary: let t denote the one with the smallest persistence. The Separatrix Persistence S_{3D} is defined for each point $x \in \ell$ as*

$$S_{3D}(x) = \max_{i=0, \dots, m} (P_{\max}(s, t) + h(x, s)). \quad (5.11)$$

Figure 5.7b shows the minimal lines of a synthetic data set that have been scaled by separatrix persistence. In contrast to the local importance I_{3D} , separatrix persistence reflects the global gestalt of the function well.

A straightforward approach in computing $S_{3D}(x)$ is to iterate over each saddle s in each level of the hierarchy $(V_i)_{i=0, \dots, m}$ and compute (5.7) and (5.11) for each point x on the separatrices of s . However, a more efficient approach is possible by exploiting that a simplification step $V_i \rightarrow V_{i+1}$ creates only local changes in the Morse-Smale complex (Section 3.5.5): only one pair of critical points gets removed with every simplification step. Hence, we compute (5.7) and (5.11) only for the separatrices that are affected in this step; and continue to the next level in the hierarchy. At V_m , we evaluate (5.7) and (5.11) for the separatrices of the (few) remaining saddles. Since separatrices are given as discrete set of links in the cell graph G , we assign the importance values only to the nodes incident to these links. An explicit sampling of separatrices is not necessary. This makes computing separatrix persistence very efficient, and can actually be done while building the hierarchy.

5.2.3 Method Overview

The input of our algorithm is a scalar field f given on any discretization that can be represented using a simplicial or cubical complex. The latter allows for a very memory efficient implementation, since node/links indices and neighborhood relations are implicitly given (Section 3.2.1). Let n denote the number of vertices of the cell complex, and let c denote the number of critical points of V_0 . The hierarchy of Morse-Smale

complexes is computed as described in Chapter 3 for the 2- and 3-dimensional case. The computation of their separatrix persistence is woven into this:

1. **Initial Gradient Field:** We use Algorithm 5 (Section 3.3) to compute the initial combinatorial gradient field V_0 . The computational effort is $O(n)$ and it allows also for a parallel computation (Section 3.3).
2. **Persistence:** Since the persistent homology of the cell complex and the Morse-Smale complex of V_0 coincide, we can compute the persistence of the critical points directly on the Morse-Smale complex itself [RWS11] with a complexity of $O(cn + c^3)$ in case of cubical complexes (Section 4.3).
3. **Hierarchy:** The hierarchy (V_i) is constructed as discussed in Section 3.5.1 and 3.5.3 for the 2- and 3-dimensional case, respectively. The computational effort depends on the topological complexity, i.e., the number of critical points and their connectivity, with a worst-case complexity of $O(n^3)$ (Section 3.5.4). However, for well-defined data the behavior is almost linear.

During hierarchization, we compute separatrix persistence (Section 5.2).

4. **Geometric embedding:** Finally, the topological structures are written out by traversing (V_i) in reverse order. The computational effort depends on the size of the structure, but is at most $O(n)$.

We found it beneficial to consider only the topological structures above an ε -persistence threshold to disregard small-scale structures. In our experiments, we set ε to 10 percent of the data range. This worked out for all of our experiments. However, this parameter depends on the application and needs to be adapted to the purpose of investigation. Since this parameter affects only the output and not the computation itself, this adaption can be easily done in a post-processing step.

Given the extraction result, the user chooses an appropriate threshold for separatrix persistence to filter noise-induced and less important (parts of) extremal lines and surfaces. After filtering by separatrix persistence, we remove small isolated lines and surfaces. Since separatrix persistence is a smooth measure, we found it beneficial to display the separation lines scaled accordingly, which elucidates the strength of a feature and allows for smooth phase-outs.

Note that different types of separation lines/surfaces share cells of the same dimension in the cell complex. This is by definition, see Section 3.2. Since these features are independent from each other, we use sparse containers to store separatrix persistence individually for separation lines and surfaces together with a reference to the corresponding cell.

Due to the combinatorial nature of our algorithm, the extracted lines and surfaces reflect the discrete nature of Ω . To obtain visually pleasing results, we apply simple heat diffusion smoothing for surfaces and Bézier curve-based smoothing for lines. Note that a strong heat diffusion smoothing yields a shrinking of the surface, and Bézier curves are not able to capture kinks. We manually adjusted the smoothing strength such that these deviations can be neglected in our investigations.

5.3 Applications of Separatrix Persistence

In the following, we present several applications of separatrix persistence for the 2- as well as 3-dimensional case.

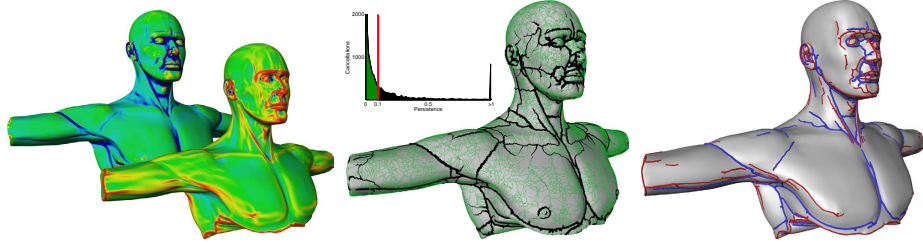


Figure 5.8: Human torso. (left) $\kappa_{max}/\kappa_{min}$ of the surface shown in front/back. (middle) Reduction of the result size by removing lines with very low persistence (green); exemplified for the minimal lines of κ_{min} . Kept lines are shown in black and scaled according to separatrix persistence. (right) Perceptually salient convex edges are shown in red, concave edges in blue.

5.3.1 Salient Edges on Surfaces

The input for the extraction of salient edges is a triangulated surface mesh and its principal curvatures κ_{max} , κ_{min} . Our method does not depend on a specific way of computing the curvatures. In fact, we found that it works well with different schemes. For the examples in this section, we used a simple curvature estimation based on normal variation.

In order to detect edges in convex regions, we start with computing the topological structures of κ_{max} . During the simplification as described in Section 3.5.1, we compute the separatrix persistence, i.e., with each cancellation step we determine the persistence of those separatrices that are removed from the gradient field (Section 5.2.1). They are hereby fixed in the hierarchy, i.e., their importance has been determined and they are added to the output. This is done until no further cancellation is possible. However, the coarsest gradient field might still contain some separatrices due to the topology of the domain. We compute their local importance and add them to the output as well.

This gives us the set of all separatrices reconnected to longer lines during the simplification and augmented with our new separatrix persistence measure. From this set we keep only the maximal lines and parts thereof which fulfill $|\kappa_{max}| > |\kappa_{min}|$, i.e., the perceptually salient maximal lines. Similarly, we get the perceptually salient minimal lines from the topological skeleton of κ_{min} which fulfill $|\kappa_{min}| > |\kappa_{max}|$.

An appropriate threshold for separatrix persistence can be found by considering the histogram of cancellations as it is shown for the torso dataset in Figure 5.8 (middle). It measures the number of cancellations over the persistence P : a very high percentage of cancellations takes place at very low persistence levels indicating the amount of noise in the data set.

Examples

Figure 5.8 exemplifies – from left to right – stages of our pipeline for extracting salient edges. Based on the principal curvatures we compute the set of salient convex and concave edges. The strength of these features is given by separatrix persistence, which is encoded for every point along the feature line. Since the result is usually rather large, we apply an automatic filtering which removes all lines that have been canceled early during topological simplification. Finally, we choose a threshold for separatrix persistence to depict the most important features. Convex edges are typically displayed

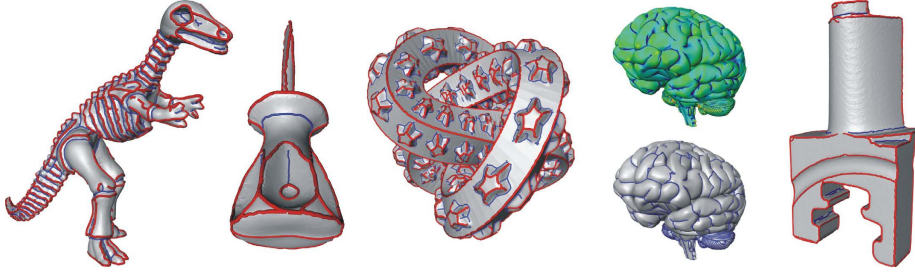


Figure 5.9: Results from left to right: dinosaur, screwdriver, knot model, plot of κ_{min} and concave edges of the brain model, blade.

in red, concave edges in blue. All models in this paper have been processed this way. Figure 5.9 shows further results.

In all practical cases, we found that the overall computation time is equally distributed between the stages. Our technique is as fast as the other tested methods: the feline data set (Figure 5.11) with its 100k triangles has been processed by all methods (see below) in under a second on the same hardware.

Comparison to Ridge/Valley-Based Methods

Since there is a large body of previous work on salient edge extraction, we feel that it is necessary to conduct a thorough comparison. In the following we compare our technique to three ridge/valley schemes. We are grateful to the respective authors for sharing their code or binaries with us:

Suggestive Contours. The SC-software package [DFR⁺] features a basic implementation of the ridge/valley definition [Thi96] applied to a curvature field computed by normal variation. The results are filtered by curvature thresholding.

Crest Lines. The method of Yoshizawa et al. [YBS05] consists of two steps: computation of a smooth curvature field by local polynom fitting and tracing of curvature extrema. Curvature computation is steered by a parameter that defines the size of the neighborhood ring, i.e., the locality of the curvature. Furthermore, the software automatically smooths the input surface. The result can be filtered by so-called ridgeness, cyclideness, or sphericalness.

Java View. This software package features the method of Hildebrandt et al. [HPW05], where a discrete shape operator is used to obtain the principal curvatures. The method explicitly incorporates the possibility to smooth the extremalities. The user has to adjust the number of smoothing steps and the step size. Results are filtered by curvature thresholding.

Robustness to Noise Triangle meshes very often contain a certain amount of noise caused by different sources. Therefore, feature extraction methods should have a certain insensitivity to noise. This is especially important when the results shall be used as input for further computations. To assess the impact of noise on the extraction results of the different methods, we use a simple generic model where we know the undisturbed

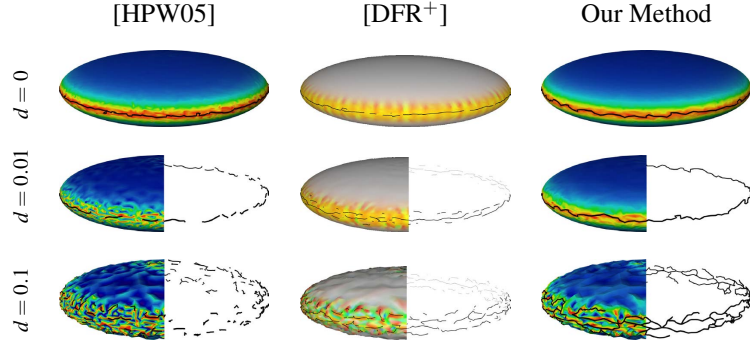


Figure 5.10: Comparison of methods regarding their robustness to noise. All methods had to deal with the raw input, i.e., any smoothing of the surface or the curvature was forbidden. Curvatures have been computed in the default way for the respective method, i.e., [HPW05] uses its discrete shape operator, while our method and the SC package use normal variation. We used a clipping plane in the last two rows to expose the feature line. As it can be seen, our method is less noise-sensitive thanks to its global nature.

result and have full control over the amount of noise. We use a sphere that is scaled in z -direction as shown in Figure 5.10. The structure we are looking for is the circum-circle in the xy -plane, which is a maximal line of κ_{max} . The objective is to reconstruct a closed circle for different amounts of noise which is added to the model by displacing the vertices into normal direction. The amount of displacement is randomized and controlled by a parameter d which scales the white noise $[0, 1] \rightarrow [0, d]$.

Figure 5.10 shows the results where the rows denote different levels of noise and the columns represent the different methods. We did not include [YBS05] in this comparison as the software did not give meaningful results for this type of model – most likely an implementational issue and not a problem of the method itself.

Already the undisturbed model poses a problem for the straightforward ridge/valley implementation of the SC package as it can be seen in the middle of the first row: the circle is a set of disconnected lines whereas our method and [HPW05] are able to extract the feature as a single closed line. As it seems, a standard ridge/valley implementation can already be affected by the very small amount of noise introduced by the discretization of the analytic model. Also, the more robust computation of curvature derivatives in [HPW05] pays off. Note, however, that our curvature computation is comparable to the one used in SC (although the surfaces are colored differently, since the SC software always uses an advanced colormap showing κ_{min} , κ_{max} while we just show κ_{max}).

In the second row, we added a small amount of noise. Both ridge/valley methods are affected by it now whereas our topological technique still captures a single closed line. We tried to push that a bit further in the last row by adding 10 times more noise. Still, our technique detects a closed structure (with a number of branches) and both ridge/valley methods are strongly affected. The reason why our topological method is much more robust against noise than ridge/valley methods lies in its global nature: local fluctuations are much less accounted for and removed early during topological simplification. In addition, the more we simplify the topological skeleton, the more it captures the overall shape of the model.

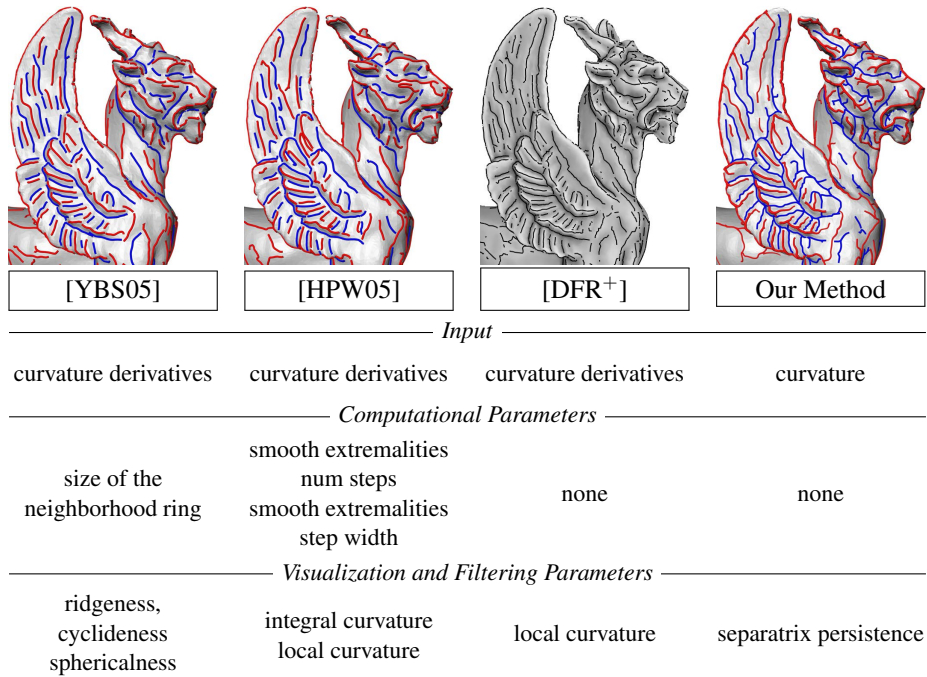


Figure 5.11: Comparison of methods regarding parameter dependence and the connectedness of the results using the feline model. Compared to ridge/valley-based methods, our technique yields qualitative similar results without derivatives and computational parameters.

We are aware of the fact that smoothing the surface or the curvature would give nicer looking results for all methods. However, “nicer looking” is not a criterion in all applications and especially not, when the results are supposed to serve as input for further computations. Smoothing may very well dislocate or otherwise alter the features in an uncontrollable way. As the results of this comparison indicate, our method is preferable in applications where noise is an issue and smoothing is not desired.

Parameter Dependence and Connectedness of Results In Figure 5.11, we used the feline model to compare all three ridge/valley methods with our technique. This model features some challenging filigree structures on the wing. All methods are able to extract the most prominent edges. However, the basic ridge/valley implementation of the SC package shows the most noisy results with a lot of disconnected lines especially next to the head. Generally, our method generates longer lines. Note, how our lines form connected subnetworks on the wing and the head. We observed this difference to the ridge/valley-based methods in all our experiments and attribute it to the global nature of our approach.

Figure 5.11 also lists the parameters of every method that had to be set up in order to achieve the shown results. It is a major advantage of our method that the computation is completely parameter-free. This makes it easier to use, especially in batch jobs where parameter adjustments for individual meshes are prohibitive. Note that the ridge/valley definition from [Thi96] itself does not impose computational parameters, since ridges and valley are well-defined mathematical constructs. The implementation in the SC

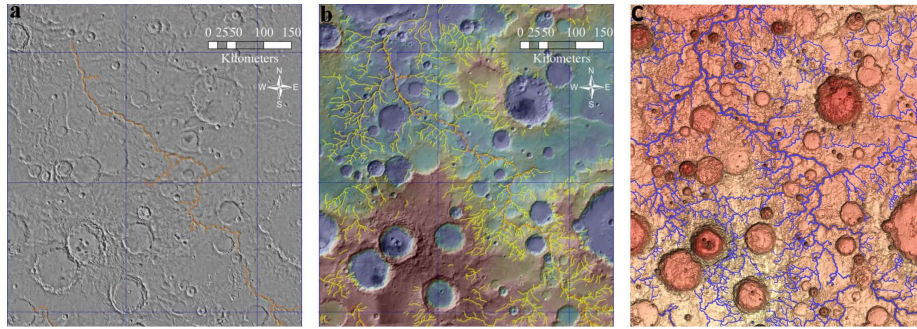


Figure 5.12: Comparison of valley networks: a) and b) show the valley networks manually mapped by Carr [Car95] and Hynek et al. [HP03] as orange and yellow lines. c) shows our automatic extraction result as blue lines (scaled by separatrix persistence).

package follows this road and does not add computational parameters. However, we find that the results of the SC package are less satisfying than those of [HPW05] and [YBS05]. Indeed, the latter two methods put more effort into an optimized computation of curvature derivatives – this is where the additional parameters come into play. The results in Figure 5.11 clearly show that this pays off compared to the basic ridge/valley approach, but it also complicates implementation and application. Our method, on the other hand, yields qualitative similar results without derivatives and computational parameters – making it easier to use and implement.

5.3.2 Valleys on the Martian Surface

We now present a second application of separatrix persistence. The objective is the detection and extraction of valley networks on the Martian surface.

The identification of valley networks and channels is an essential tool for geomorphological interpretations of the fluvial, glacial and volcanic history of Mars. While the creation of valley networks by erosion is an accepted hypothesis, the flow of water as the sole cause has recently been put into question [Ble10]. To investigate the origin of the networks, their detailed properties have to be mapped at a global scale. In previous attempts of computer-generated global mapping, manual verification of the results was necessary [LS09]. Herein, we automatically extract valley networks in terms of extremal lines and compare the extraction results with manually mapped ones that were published in [Car95, HP03].

Elevation maps of the Martian surface have a preponderance of craters scattered on them. This complicates a direct application of the topological analysis. In terms of persistence the craters correspond in general to dominant minima. The incident separatrices are therefore also dominate. However, we can use the topological analysis to circumvent this problem by masking out the craters.

Given an elevation map, we first compute the hierarchy of topological features as in Section 3.5.1. In this first step, we are only interested in the minima. Among other features, the minima represent craters. The data analyst chooses a level of detail such that all craters contain at least one minimum. The set of minima are used as seed points for a simple flooding algorithm. For each seed point, the flooding is done until the corresponding crater is covered. Given this binary mask, we can now start to extract all extremal lines. We compute separatrix persistence (Section 5.2.1) for the masked

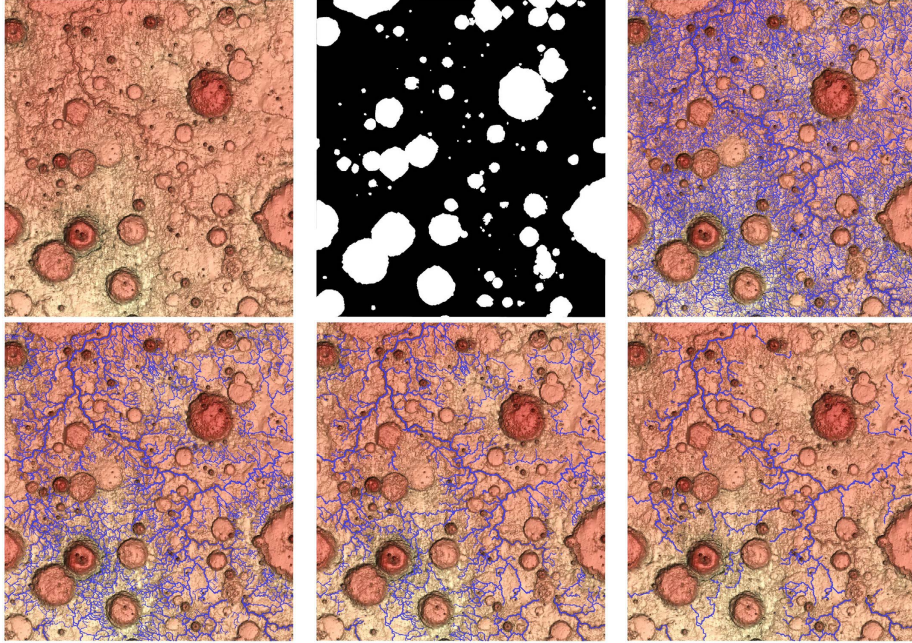


Figure 5.13: Different levels of detail of extremal lines: For a given elevation map, a binary mask was computed to mask the craters. Four levels of the hierarchy are depicted – the initial topological structures (where all extremal lines are present) and three less detailed levels are shown. The width of the extremal lines is scaled by separatrix persistence.

elevation map. In this step, we are interested in the minimal lines and their importance.

We applied our new method to Mars Orbiter Laser Alimeter (MOLA) data with a resolution of 128 pixels per degree. Specifically, we concentrated on the region that was already investigated by Carr [Car95] and Hynek et al. [HP03]. This allows for a comparison of their manual mapping with our automatically computed hierarchy. As can be seen in Figure 5.12, the extremal lines extracted by our method cover most of the lines manually mapped by Hynek et al. [HP03]. This indicates that separatrix persistence assigns highly important-values to erosional structures. However, there are also additional lines that were not mapped by them. Figure 5.13 depicts four levels of the hierarchical representation of the masked elevation map. While in the initial level all extremal lines were present, only the most dominant lines remain in the last levels. The central channel is clearly the most dominant structure.

5.3.3 Computer Tomography

Computer tomography (CT) scans usually suffer from a large amount of noise which challenges the extraction of the extremal structures therein. We computed separatrix persistence as described in Section 5.2.2 and applied it to two data sets: a CT-scan of an aneurism and a bonsai tree. As discussed in the Sections 1.1 and 5.1, extremal structures can be extracted by a local or a global analysis, we compared our extraction result of the bonsai data set to the concept of Height Ridges [PS08]. The computation were done on an Intel Xeon E31225 (3.1GHz) CPU and 16 GB RAM. The running time

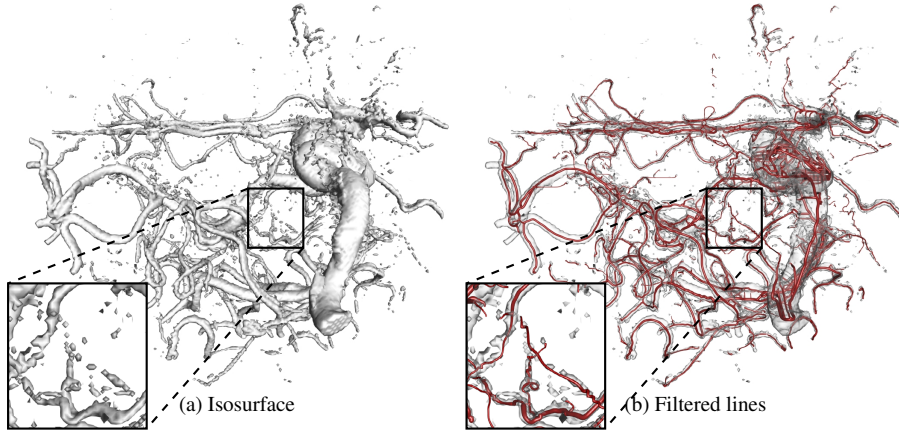


Figure 5.14: Extraction of maximal lines in a CT-scan of an aneurism using separatrix persistence.

of the complete pipeline as described in Section 5.2.3 was 4 minutes for the aneurism and 15 minutes for the bonsai data set. Both data sets are given on a uniform 256^3 grid.

The aneurism consists of many filigree structures describing its blood vessels. Due to the noise, these vessels are interrupted and represented by scattered surfaces as can be seen in the isosurface shown in Figure 5.14a. We applied our method with the goal to extract blood vessels as maximal lines. Figure 5.14b shows the smoothed result filtered and scaled by separatrix persistence. Even in the presence of noise, connected blood vessels are robustly extracted.

Figures 5.15 and 5.16 show the results of [PS08] and our method, respectively, applied to a CT-scan of a Bonsai tree. The objective in this data set is the extraction of the tree-skeleton – the trunk with all its branches. It appears as lines of maximal intensity in the CT-scan.

To extract the ridge lines, we first had to smooth this data set using a Gaussian filter. This reduced the noise level such that we were able to extract a meaningful result. The ridge definition alone yields a wealth of scattered lines, where the trunk and the tree are not identifiable, see Figure 5.15a. Peikert and Sadlo [PS08] proposed a filter criterion F_α based on the angle α between the ridge line and the gradient. Typically, α varies between 5° and 60° . We applied this filter criterion F_{45} with $\alpha = 45^\circ$.

While this reduced the complexity of the extraction result, the trunk and its branches are still not detectable, see Figure 5.15b. We restricted the ridge computation to regions with $f > 55$. Using this restriction, the overall structure of the tree becomes visible. However, the branches and the trunk are still represented by a scattered set of lines, see Figure 5.15c.

We applied all topological computations to the original data set (no smoothing). The result is shown in Figure 5.16a, where we show *all* maximal lines scaled by separatrix persistence. While the overall scenery is quite complex, the structure of the tree is already identifiable. Figure 5.16b shows the result after filtering by separatrix persistence. Due to the combinatorial nature of our method, the maximal lines follow the grid structure as it can be seen in the close-up of Figure 5.16b. We smoothed the maximal lines to obtain a visually pleasing result, see Figure 5.16c. The smoothing introduced only slight deviations in the geometric embedding. Note how the trunk and its branches

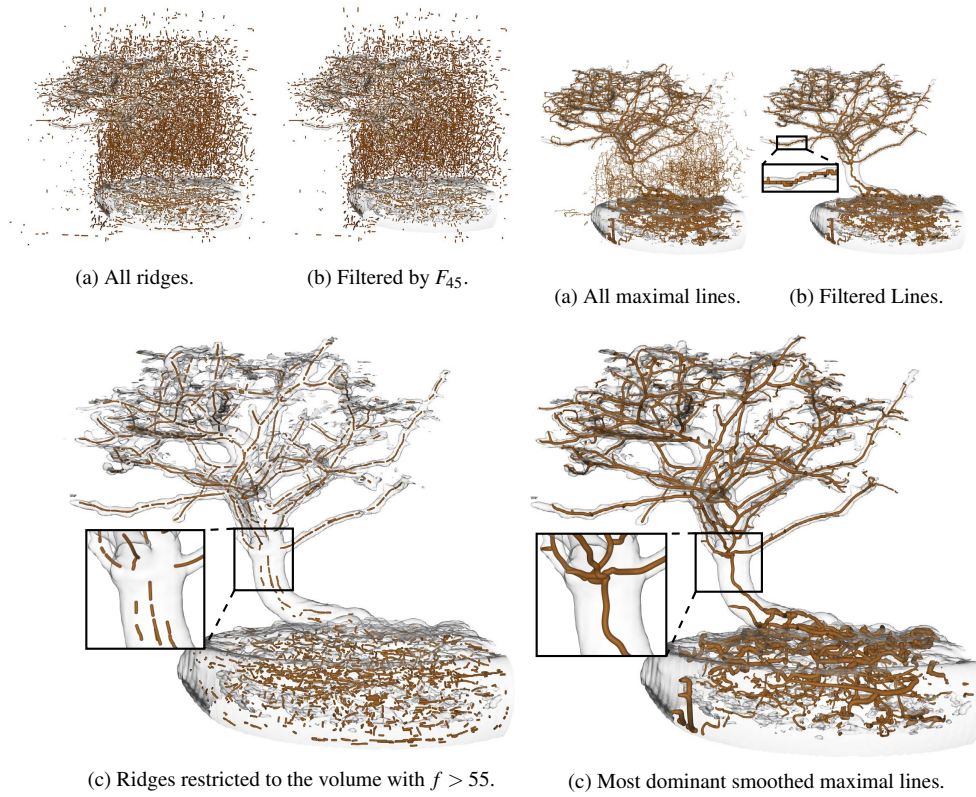


Figure 5.15: Height ridges of the Bonsai data set.

Figure 5.16: Maximal lines of the Bonsai data set.

are nicely represented as a connected network in contrast to the ridge extraction result.

5.3.4 Scalar Quantities in Fluid Dynamics

We computed separatrix persistence as described in Section 5.2.2 and applied it to two data sets from the fluid dynamics domain: In the first experiment, we extracted the most dominant extremal lines and surfaces from a scalar quantity, i.e., the Q -criterion, derived from fluid dynamics simulation of a Kármán vortex street. Similar as for the bonsai example in Section 5.3.3, we compared our result to the method of Peikert and Sadle [PS08]. The second experiment focuses on the most dominant parts of separation surfaces in an FTLE field. The computation were done on an Intel Xeon E31225 (3.1GHz) CPU and 16 GB RAM. The running time of the complete pipeline as described in Section 5.2.3 was 3 minutes for the vortex street and 10 minutes for the FTLE field.

Figure 5.17 demonstrates the results of our method applied to a scalar quantity derived from a flow behind a cylinder. The data set was provided by Bernd R. Noack (TU Berlin) from a direct numerical Navier Stokes simulation by Gerd Mutschke (FZ Rossendorf). It resolves the so called “mode B” of the 3D cylinder wake at a Reynolds number of 300 and a spanwise wavelength of 1 diameter. The data is provided on a

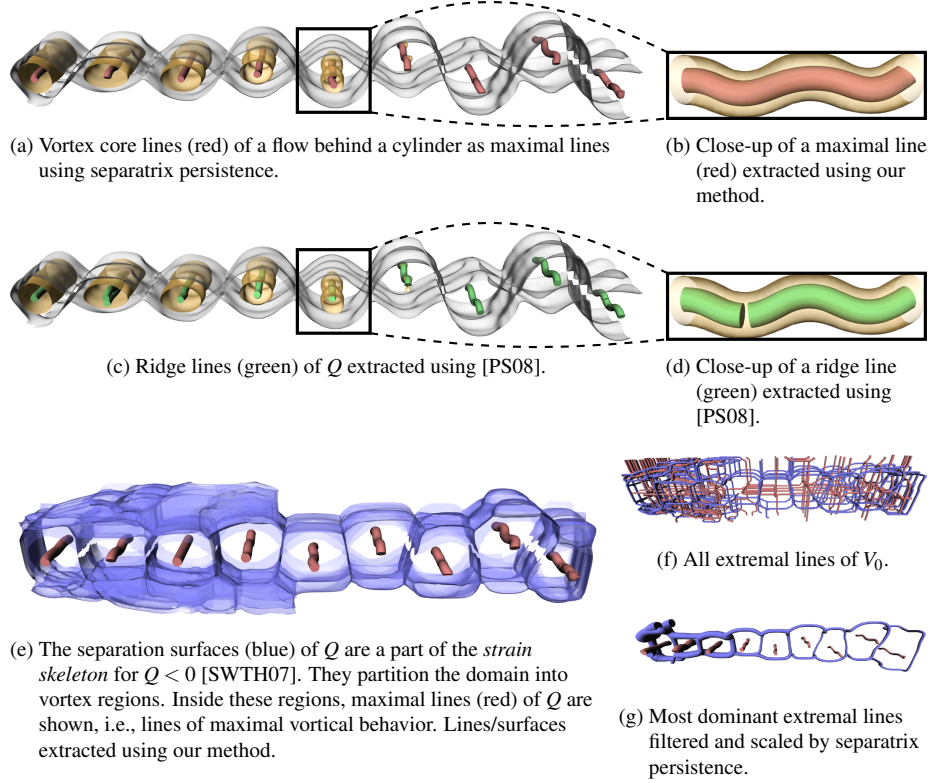


Figure 5.17: 3D unsteady flow behind a cylinder. Shown are extremal features of the Q -criterion for $t = \pi$. The gray isosurface depicts the zero-level of Q while the level $Q = 2.7$ is illustrated as yellow isosurface.

$265 \times 337 \times 65$ curvilinear grid as a low-dimensional Galerkin model. The examined time range is $[0, 2\pi]$. The flow exhibits periodic vortex shedding leading to the well known von Kármán vortex street [ZFN⁺95]. This phenomenon plays an important role in many industrial applications such as mixing in heat exchangers or mass flow measurements with vortex counters. However, this vortex shedding can lead to undesirable periodic forces on obstacles such as chimneys, buildings, bridges and submarine towers.

We analyze the Q -criterion [Hun87] of this flow, which is a derived scalar field that allows to distinguish between vortex ($Q > 0$) and strain ($Q < 0$) behavior. The latter measures the amount of stretching and folding which drives mixing to occur. As pointed out by Sahner et al. [SWTH07], the minimal points/lines/surfaces of Q represent the *strain skeleton*, while the maximal features of Q denote the *vortex skeleton*.

Figures 5.17f-g provide a comparison between the unfiltered extremal lines and the lines filtered and scaled by separatrix persistence. Minimal lines are shown in blue, maximal lines in red. This exemplifies that separatrix persistence is able to reveal the most dominant features. We additionally applied a derived filter criterion here: the variance of separatrix persistence along a line. The idea is to favor lines that stay in the center of a vortex, i.e., that have a rather constant Q -value and therefore a rather constant separation strength.

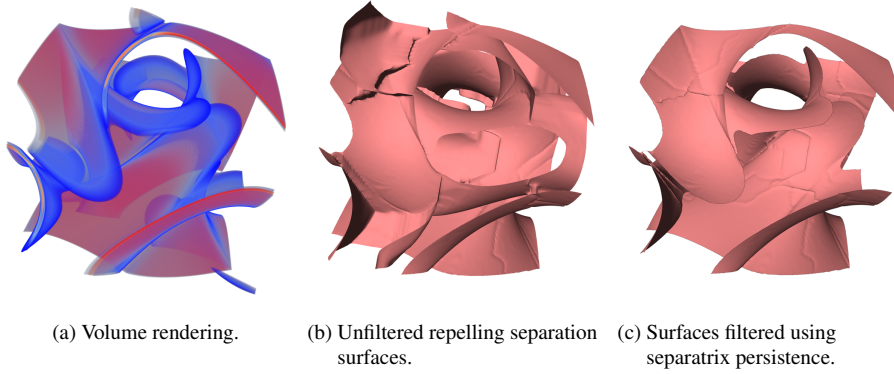


Figure 5.18: FTLE of the ABC Flow. Filtering the separation surfaces of the Morse-Smale complex using separatrix persistence reveals the most dominant surfaces of maximal FTLE value.

The most dominant maximal lines and ridges of Q are shown in Figures 5.17a and 5.17c, respectively. The ridge lines are filtered by the F_{45} filter [PS08]. We additionally removed small isolated lines. Both extraction methods yield qualitatively very similar results. However, the close-ups in Figures 5.17b and 5.17d reveal an important difference of the two approaches: The topological approach gives long, fully connected lines (Figure 5.17b). In contrast, ridge lines are often split into several smaller parts in this data set (Figure 5.17d). This is due to the fact that ridge lines are local features, i.e., it is locally decided whether or not a point is on a ridge or not. Due to numerical instabilities or noise, some of the local decisions along a ridge line may produce a “miss”, which then leads to disconnected results. This cannot happen for the topological approach, since separatrices are global features. On the other hand, ridge lines do not suffer from deviations due to smoothing.

Figure 5.17e shows the separation surfaces emanating at the 2-saddles of Q restricted to $Q < 0$. Following Sahner et al. [SWTH07], this provides a partition of the domain into vortex regions, which is nicely confirmed by the shown vortex core lines in the center of each of these regions.

The Finite Time Lyapunov Exponent (FTLE) [Hal01] is a scalar field that describes the separation of particles in a flow: high FTLE values indicate a strong separation, i.e., neighboring particles diverge from each other during integration. Hence, one is interested in finding surfaces of maximal FTLE value. We computed FTLE for the well-known ABC flow from [Hal01]. This is an analytic flow given on a uniform 256^3 grid and the numerically computed FTLE field exhibits almost no noise. We include this example to showcase that separatrix persistence is not only useful for filtering noise-induced structures, but also to find the most dominant parts of features, i.e., the parts with the highest feature strength. Figure 5.18a shows a volume rendering of this scalar field. The transfer function has been chosen to show only regions of very high FTLE values. Figure 5.18b shows all separation surfaces emanating at the 1-saddles. We find the most dominant parts of these surfaces by filtering them using separatrix persistence, as shown in Figure 5.18c.

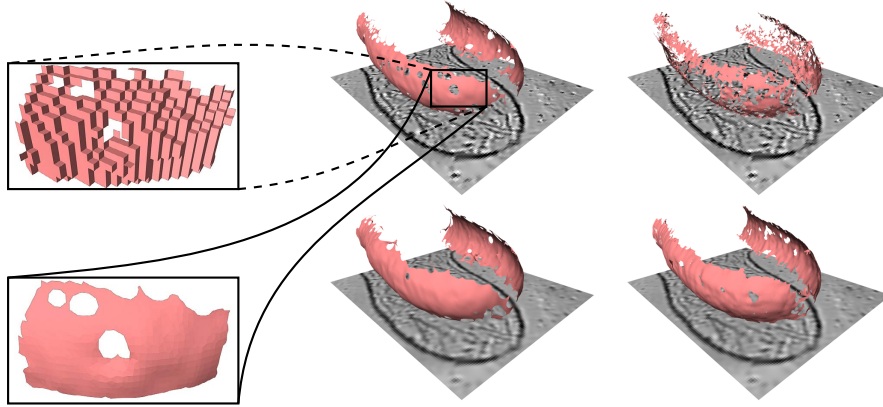


Figure 5.19: Discrete (top) and smooth (bottom) representation of a subregion of the cell membrane.

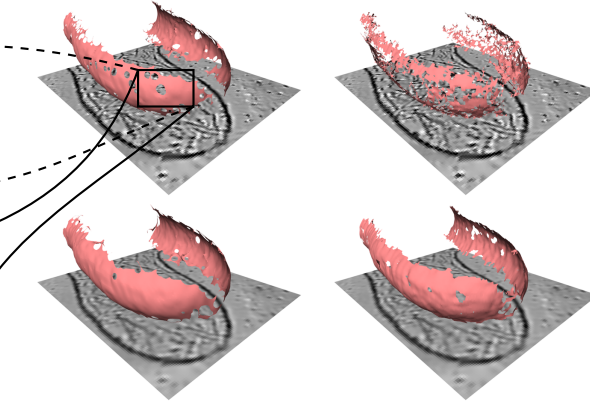


Figure 5.20: Extraction of a cell membrane in a cryo-electron tomogram using separatrix persistence (left column) and ridge/valley definition (right column). The top row shows the original data. The bottom row shows the results after Gaussian smoothing.

5.3.5 Cell Biology

Cryo-electron tomography allows to visualize sub-cellular structures such as cell membranes. This imaging technique suffers from a very low signal-to-noise ratio and artifacts arising from incomplete information (“missing wedge”), which makes an automated extraction of these structures very challenging [Fra06]. We applied our method to a sub-tomogram ($124 \times 154 \times 47$) of a *Dictyostelium discoideum* cell [RGH⁺12b] with the goal to extract the cell membrane as the dominant parts of the combinatorial separation surfaces. We compared our result with ridge surfaces computed using the technique of Peikert and Sadlo [PS08]. The computation were done on an Intel Xeon E31225 (3.1GHz) CPU and 16 GB RAM. The running time of the complete pipeline as described in Section 5.2.3 was 7 minutes for this data set.

The sub-tomogram shows the so-called *Filopodium* – a finger-like extension of the cell. Figure 5.20 (left, upper row) shows the result after filtering using separatrix persistence and smoothing the surface using heat diffusion (see Figure 5.19 for the effect of the surface fairing). Although some holes within the membrane occur, its overall shape is well recovered. As described by Rigort et al. [RGH⁺12b], the tomogram was already filtered using non-local means [BCM05]. However, this filtering is not sufficient for the extraction of ridge surfaces as shown in Figure 5.20 (right, upper row): the membrane is only represented by a scattered set of small surface pieces.

The remaining noise level challenges the ridge computation, in contrast to the topological approach. We applied Gaussian smoothing to lower the noise level further. The bottom row of Figure 5.20 shows the results for the smoothed version. Both extraction approaches benefit from this smoothing step. The cell membrane is almost closed, only few holes remain.

Chapter 6

Discussion

In the following, we discuss the applicability of the algorithmic framework presented in Chapter 3. Since the overall setting is combinatorial, we especially discuss the relationship and differences to the smooth setting. We also investigate the limitations of our new importance measure separatrix persistence introduced in Chapter 5, and the extension of our framework to higher dimensional data.

6.1 Smooth vs. Discrete Topological Structures

In Chapter 3, we presented algorithms that compute the topological structures – the critical points, and the separation lines and -surfaces – within a scalar field. These structures are given as a subset of links and nodes in the graph. Figure 6.1 shows these structures of the electrostatic field around a benzene molecule. This data set was

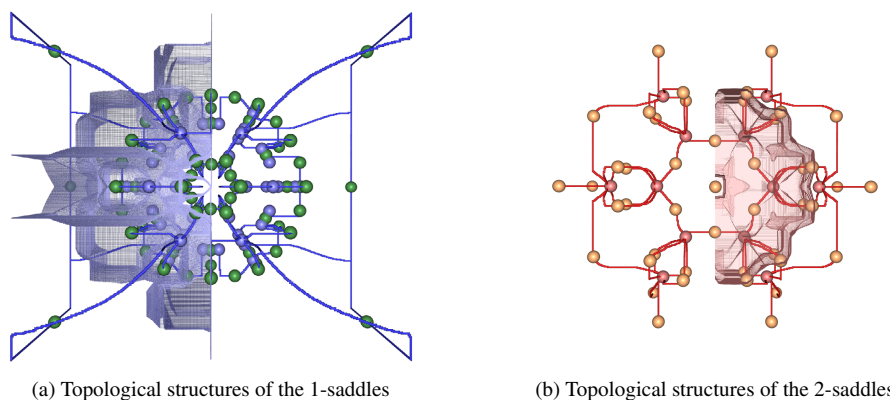


Figure 6.1: Separation lines and surfaces of the electrostatic field of a benzene molecule for V_{m-85} with 181 critical points. (a) shows the minimal structures: the 48 minima (blue spheres), 78 1-saddles (green spheres), 0-separation lines (blue lines) and the ascending manifold of the 1-saddles (blue surface). (b) shows the maximal structures: the 12 maxima (red spheres), 43 2-saddles (yellow spheres), 2-separation lines (red lines) and the descending manifolds of the 2-saddles (red surface). Due to symmetry, only one half of the separation surfaces is shown.

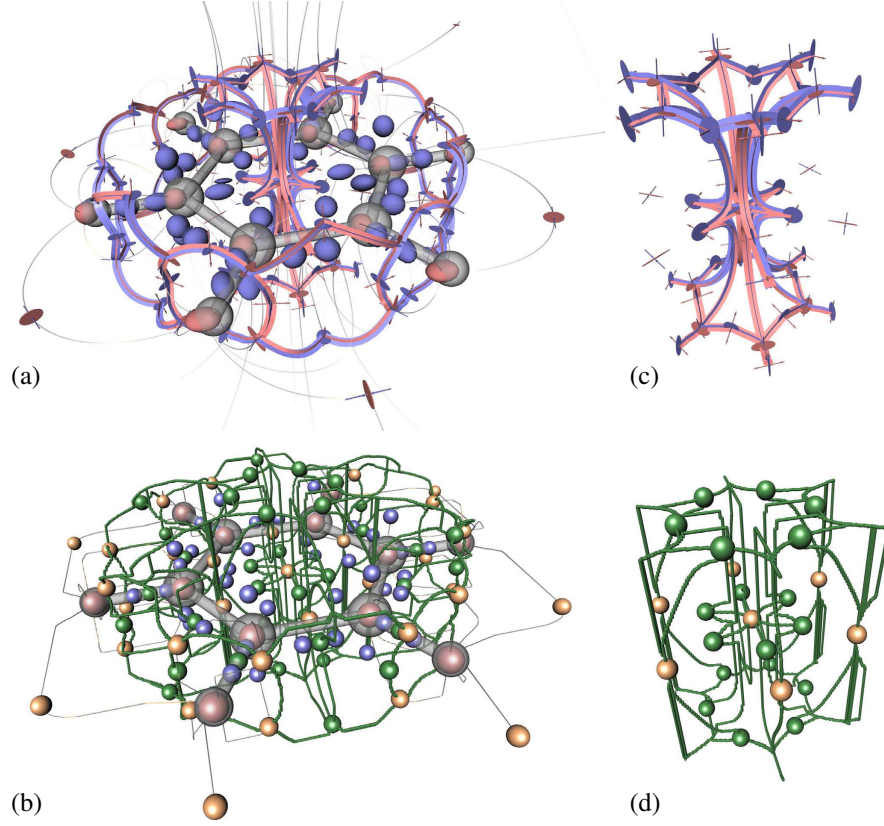


Figure 6.2: Comparison of combinatorial and continuous extremal structures for the electrostatic field around a benzene molecule. (a) shows smooth extremal structures extracted as in [TWHS03a]. The minima and the maxima are depicted as blue and red spheres while the 1- and 2-saddles are shown as blue and red disks respectively. The 1-separation lines are shown as blue-red stripes. Gray illuminated lines represent streamlines emanating from the saddles. (b) shows combinatorial extremal structures. The minima, 1- and 2-saddles, and the maxima are represented by blue, green, yellow and red spheres respectively. The 1-separation lines are shown as green lines. Gray illuminated lines depict the 2-separation lines emanating from the 2-saddles. (c) and (d) show a close-up of the center 2-saddle with its surrounding 1-saddles and the corresponding 1-separation lines for the smooth and combinatorial case. Gray surfaces depict the carbon and the hydrogen atoms and their bonds.

already analyzed in [TWHS03a] using numerical methods. Since sampling introduce spurious critical points, we chose the hierarchy level V_{m-85} such that we got the same number of 1- and 2-saddles as in [TWHS03a]. The electrostatic field was sampled on a 401^3 regular grid using the fractional charges method described by Stalling et al. [SS96].

As can be seen in Figure 6.1, the regularity of the underlying data set has been perfectly captured. This poses a challenge for numerical algorithms, since guarantees about finding all critical points can usually not be given. A side-by-side comparison of the continuous and the combinatorial extraction results is shown in Figure 6.2. The

close-ups in Figures 6.2 (c) and (d) reveal a high qualitative similarity between both versions.

We make the following observations: Numerical algorithms require a larger number of parameters, which are often difficult to choose. In this example, the continuous version misses some 1-separation lines (saddle connectors), since a certain maximal number of integration steps had to be chosen for the extraction algorithm [TWHS03a]. Of course, we could have changed that parameter and re-run the algorithm by Theisel et al. [TWHS03a], but this still would not make it a proofably watertight case. The combinatorial algorithm, on the other hand, captures all connectors by design.

On the other hand, the smooth nature of the flow is better communicated to a viewer in the continuous version. For such visualization purposes, the classic continuous methods are preferable over the combinatorial ones. In contrast to the numerically computed separation lines (Figures 6.2a and 6.2c), the geometric embedding of the combinatorial separation lines shown in Figures 6.2b and 6.2d clearly reflects the structure of the underlying grid.

The main reason for these differences in the geometric embedding is the combinatorial representation of the gradient direction in terms of the directions provided by the grid. We use Algorithm 5 proposed by Robins et al. [RWS11] to compute the combinatorial gradient field. Herein, the grid direction with the steepest descent is chosen to approximate the continuous gradient. One could argue that a refinement of the grid would reduce these differences, and the combinatorial structures would then converge to the continuous ones. Unfortunately, this is not the case. When one follows the combinatorial gradient, the quantization error of the topological structures can accumulate to a large value.

In the following, we investigate this behavior in detail.

6.1.1 The Steepest Descent

For simplicity, we now assume that a cubical complex is given. These complexes are usually employed if image data are analyzed. However, we will also provide some comments regarding simplicial complexes.

The separatrices of a combinatorial gradient V are alternating paths in the cell graph G with respect to V . Intuitively, the links in V should therefore reflect the direction of the continuous gradient of the input data. However, at a given vertex u , there are only a constant number of directions representable by the links of G . This implies that the continuous gradient can only be represented in a quantized way. Loosely speaking, the gradient direction is snapped to the links of the graph.

Therefore, the combinatorial gradient differs from the continuous gradient not only by a sampling error, but also by a quantization error. The sampling error can be decreased using a denser sampling. However, this is not necessarily the case for the quantization error.

The steepest descent strategy for the link selection in Line 16 of Algorithm 5 suffers from this quantization artifact. Suppose that the exact gradient is almost constant in a region K and points 'North-North-East' as shown in Figure 6.3. At any given vertex in K the steepest descent direction is therefore always 'North' – independent of the resolution used to sample the exact gradient. Any exact separatrix passing through K is thereby approximated by a straight line going 'North'.

The same behavior can be also observed for simplicial complexes. A triangulation of a surface is usually designed in a way such that it represents the local details of a surface well. Very often such triangulations are also locally structured which yields the

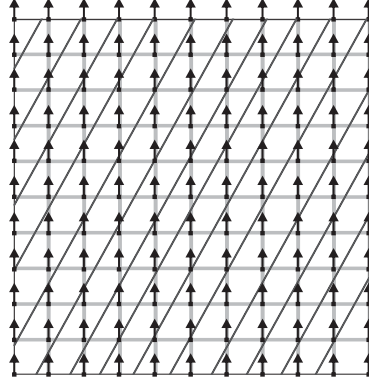


Figure 6.3: Illustration of the steepest descent of a locally constant gradient (black diagonal lines). The gradient is projected on a uniform grid (gray). The steepest descent at the vertices of the grid (black arrows) always points north.

same problem as with the cubical complex. Only a limited number of directions are available.

A possible way to resolve this problem is to increase the number of directions such that the link selection corresponds to the continuous gradient. For a given simplicial complex, the number of available directions can be increased by employing a remeshing. In many applications, however, a lot of effort is put into the design of a triangulation and a subsequent altering of it is not desired. If a remeshing is employed, the resulting triangulation must be locally unstructured, i.e., no specific direction should be given by the edges of the triangulation.

The cubical complex is implicitly given which enables the treatment of very large data. In contrast, the simplicial complex is explicitly given; the cell and their adjacency require an explicit storage. A direct translation of the cubical complex into a simplicial complex would increase the memory consumption by several factors. This might exceed the available memory on commodity hardware.

In the following, we will propose and investigate an alternative approach to dealing with the quantization error: We choose the links adjacent to a vertex u in a probabilistic fashion. Since we cannot represent the (continuous) gradient direction exactly, we pick a link according to a random variable X_u . For a cubical complex, the probability mass function g of X_u is defined by the scalar field f and the width and height of each pixel. The basic idea is to design g such that the expected value of X_u corresponds to the (continuous) gradient direction at u .

These random variables are independent. Assuming that in this setting the law of large numbers [Ber13] is applicable, a path following this probabilistic combinatorial gradient will therefore almost surely proceed in the direction of the (continuous) gradient when the grid is refined. We now provide an idea how such a probabilistic link selection can be designed for cubical complexes in the 2-dimensional case.

6.1.2 Probabilistic Steepest Descent

The probabilistic approach discussed in the following assumes a cubical 2-dimensional complex. The idea is to replace the link selection in Algorithm 5. We choose the direction (Line 16 of Algorithm 5, page 45) from the set of admissible links L in a

probabilistic fashion instead of choosing the locally steepest descent.

The index of the links in L is either always 1 or always 0 (Line 14 of Algorithm 5). Perhaps surprisingly, it suffices to choose the links of index 0 appropriately. The order in which the links of index 1 are chosen has no effect. They are uniquely defined once the links of index 0 and the saddle points have been selected. This fact motivated the construction method of a combinatorial gradient by Lewiner et al. [LLT03]. In the following, we therefore assume that L contains only links of index 0.

For a given vertex $u^0 \in N$, the link selection strategy is given by a random variable X_u . The value of this random variable is always a link in L . We now define a probability mass function $g : L \rightarrow [0, 1]$ for X_u such that the expected value of X_u is collinear to the (continuous) gradient at u .

We assume (without loss of generality) that the (continuous) gradient points North-East, i.e., $\nabla f(u) = (f_x, f_y)$ with $f_x, f_y \geq 0$. Furthermore, the width of the current pixel is denoted by w and its height by h . The set L thereby consists of the directions $(0, h)$ and $(w, 0)$.

To simplify notation, we refer to $g((w, 0))$ by λ . Since g is a probability mass function, we have $g((0, h)) = 1 - \lambda$. The expected direction $E(X_u)$ is now given by

$$E(X_u) = (1 - \lambda) \begin{pmatrix} 0 \\ h \end{pmatrix} + \lambda \begin{pmatrix} w \\ 0 \end{pmatrix} = \begin{pmatrix} \lambda w \\ (1 - \lambda)h \end{pmatrix}. \quad (6.1)$$

Since $E(X_u)$ should be collinear to $\nabla f(u) = (f_x, f_y)$, the following condition must hold

$$\det \begin{pmatrix} \lambda w & f_x \\ (1 - \lambda)h & f_y \end{pmatrix} = 0. \quad (6.2)$$

This yields

$$g((w, 0)) = \frac{hf_x}{wf_y + hf_x} \quad \text{and} \quad g((0, h)) = \frac{wf_y}{wf_y + hf_x}. \quad (6.3)$$

Since $\nabla f(u)$ is not directly available, we approximate it using finite differences:

$$f_x \approx \frac{f(u + (w, 0)) - f(u)}{w} \quad \text{and} \quad f_y \approx \frac{f(u + (0, h)) - f(u)}{h}. \quad (6.4)$$

Denoting the height difference $f(u + (w, 0)) - f(u)$ by W and $f(u + (0, h)) - f(u)$ by H , and inserting (6.4) into (6.3) yields the final probability mass function g in terms of f , w and h :

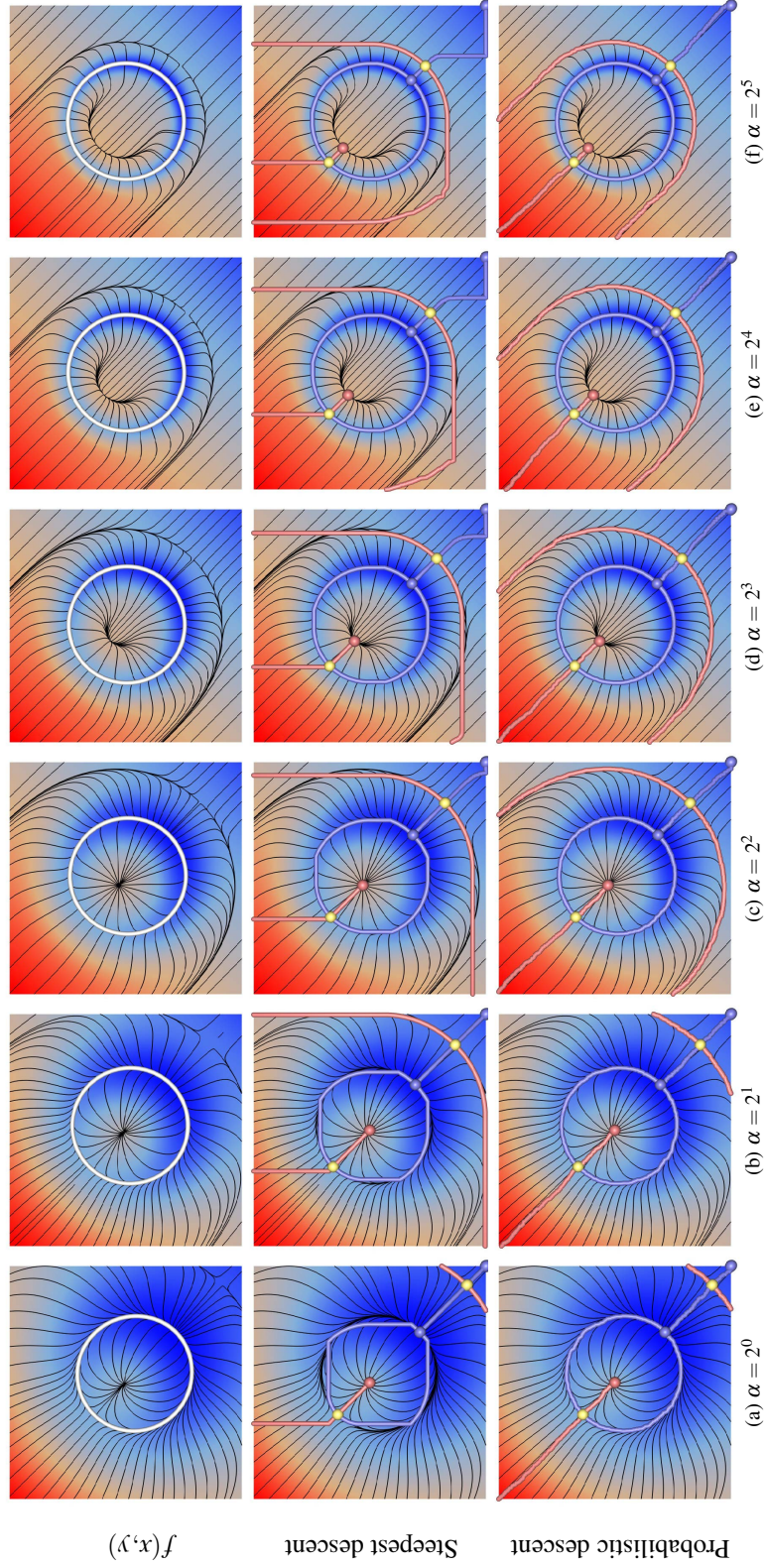
$$g((w, 0)) = \frac{h^2 W}{w^2 H + h^2 W} \quad \text{and} \quad g((0, h)) = \frac{w^2 H}{w^2 H + h^2 W}. \quad (6.5)$$

Note that, in practice, L may consist of more than two links due to the sampling of f . Each link in L is therefore assigned the height difference weighted by the squared length of the dual link. Its probability is then given by the normalized value with respect to the other links in L .

6.1.3 Empirical Convergence

We now evaluate how the above probabilistic idea performs compared to the steepest descent strategy. Let $\Omega = [-2, 2]^2$ and $\alpha \in \mathbb{R}^+$. The function $f : \Omega \rightarrow \mathbb{R}$ is given as

$$f(x, y) = -e^{-\alpha(\sqrt{x^2 + y^2} - 1)^2} - 0.3(x + y). \quad (6.6)$$



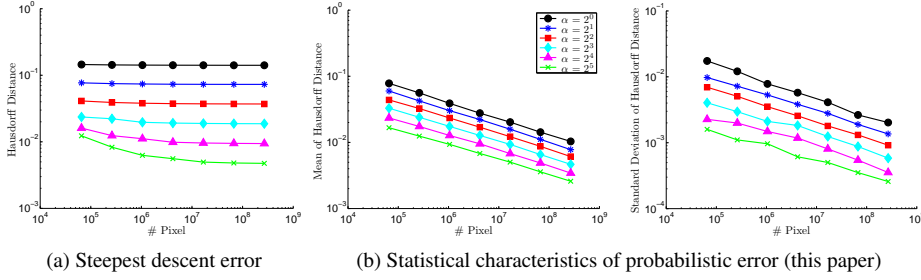


Figure 6.5: The analytic example f (6.6). (a) and (b) show the Hausdorff distance of the approximated center circle to the reference circle for different choices of α and increasing resolution. (a) shows the evolution of the error using the steepest descent approach. (b) shows two statistical characteristics of the error of the probabilistic approach.

The function f describes a circle engraved on a tilted plane. The sharpness of this circle is defined by the parameter α . For $\alpha \rightarrow \infty$, the circle becomes arbitrary sharp. For $\alpha \rightarrow 0$, f gets flattened. Varying α allows us to simulate smooth as well as sharp features appearing in many applications. An illustration of f sampled on a 2048^2 grid for different choices of α is given in the first row of Figure 6.4. Integral lines of the continuous gradient ∇f are depicted by black lines using the dual streamline seeding technique by Rosanwo et al. [RPP⁺09]. Converging integral lines indicate thereby the existence of a separatrix. In the following paragraphs, we choose the engraved circle as a reference feature. For illustration, it is shown as a white circle line in the first row of Figure 6.4.

We applied Algorithm 5 using the steepest descent version as well as the probabilistic version to construct a combinatorial gradient of f for different choices of α . The resulting Morse-Smale complexes of the steepest descent version are shown in the second row of Figure 6.4. Our reference feature – the white circle – is visually well recovered for large values of α , which confirms also the extraction results in Section 5.3 and of recently proposed methods [CCL03, KRHH11]. In many applications, the desired features are sufficiently sharp.

However, deviations from the reference circle become visible if the feature gets smooth, i.e., for small choices of α . The steepest descent version of Algorithm 5 is not able to recover the circle for $\alpha = 2^0$ and $\alpha = 2^1$. The probabilistic approach, in contrast, is able to recover the circle for all choices of α . The resulting Morse-Smale complexes are shown in the third row of Figure 6.4.

To quantify the approximation error, we measured the Hausdorff distance [Hau14] of the reference circle to the approximated circle. Figure 6.5a shows the approximation error for Algorithm 5 using the steepest descent strategy in a log-log plot. Although the extraction result looked visually reasonable for large α -values, there is no convergence for any α . The Hausdorff distance to the reference circle is not decreasing if a finer grid is employed.

For the probabilistic approach, we did 200 runs of Algorithm 5. Figure 6.5b shows the mean value of the Hausdorff distance and its standard deviation. Both quantities are converging to zero. Hence, the sampling as well as the quantization error are reduced when a finer grid is employed. However, it needs to be noted that the approximation error for very sharp features ($\alpha = 2^5$) is slightly larger compared to the steepest descent

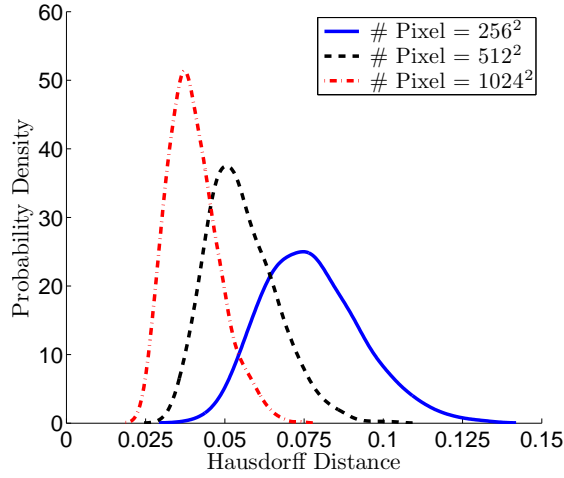


Figure 6.6: Distribution of Hausdorff distance error. The blue, black and red curves show the estimated probability density function of the Hausdorff distance between the center circle and the reference circle (as shown in Figure 6.4) for different resolutions.

strategy when coarse grids are used. It is also worth mentioning that the convergence speed is not optimal since we employed uniform refinements. Increasing the efficiency may be achieved by developing an adaptive grid refinement approach. This may be particularly effective since a high resolution is only needed in the vicinity of the separatrices.

In Figure 6.6, we plotted a kernel density estimate [BGK10] of the probability density of the Hausdorff distance error for different resolutions. We applied Algorithm 5 using the probabilistic approach 1000 times to obtain a sufficient number of samples for the density estimate. The overall shape of the densities suggests a sequence of binomial error distributions that converge to a normal distribution.

To demonstrate that the probabilistic approach is convergent for general smooth functions, we considered a set of smooth functions generated by the expression

$$\sum_{m,n=1}^2 \left(X_{m,n}^{(1)} \sin(mx) + X_{m,n}^{(2)} \cos(mx) \right) \left(X_{m,n}^{(3)} \sin(ny) + X_{m,n}^{(4)} \cos(ny) \right), \quad (6.7)$$

where the $X_{m,n}^{(j)}$'s are random variables uniformly distributed in $[-1, 1]$. This expression is now evaluated on the domain $[-\pi, \pi]^2$ discretized using two different grid resolutions. We selected four representatives of this set of functions and applied the steepest descent and our probabilistic version of Algorithm 5. The resulting separatrices are shown in Figure 6.7.

It is apparent that the steepest descent approach does not yield separatrices with a correct embedding. The proposed probabilistic link selection strategy, however, visually converges to the correct solution. We applied this idea to a much larger number of such functions, and always observed the above behavior.

We can conclude that a probabilistic extension of Algorithm 5 yields combinatorial gradients whose separatrices converge to their continuous counterpart when the grid resolution is increased. We can therefore benefit from the simplicity and robustness of discrete Morse theory, and still obtain topological structures that converge to

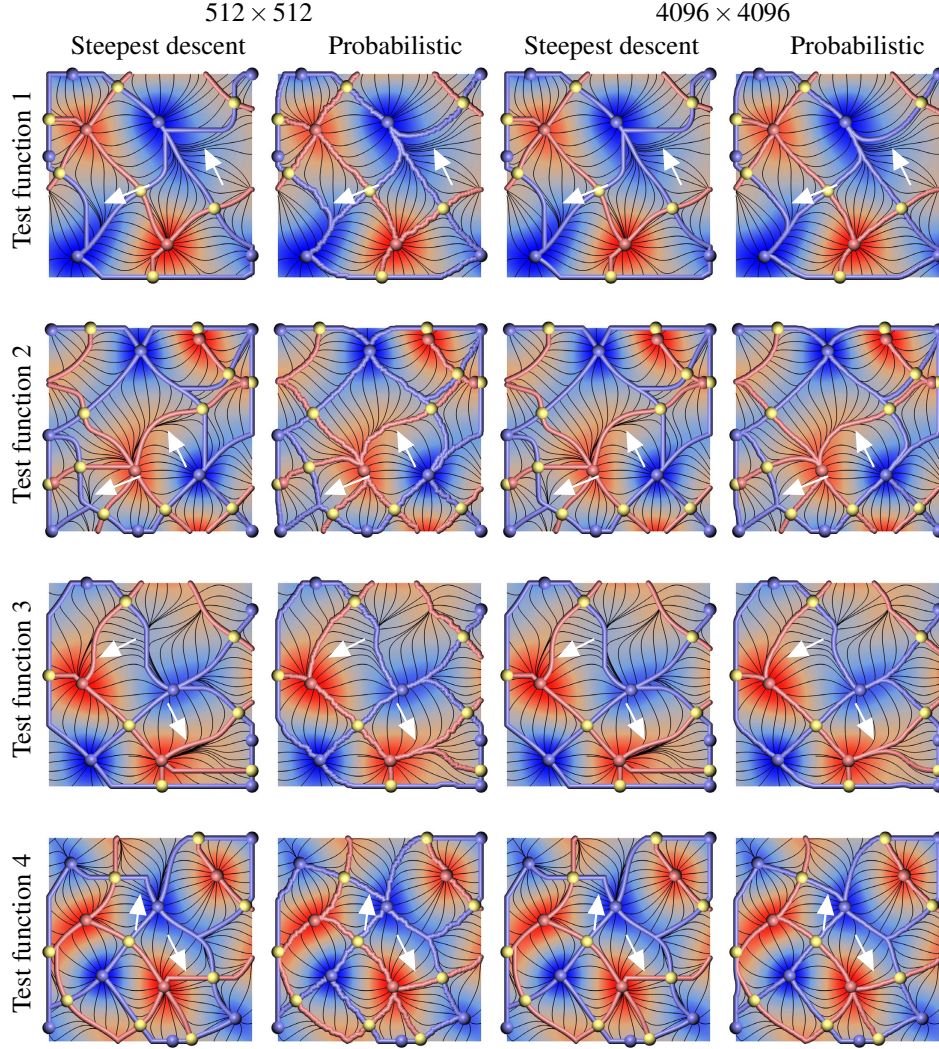


Figure 6.7: Comparison of the steepest descent and the probabilistic approach using generic functions. The four rows show four different functions randomly generated using expression (6.7). Red denotes a high function value, while blue denotes a low value. Black lines depict integral lines of their gradient. The left two columns show the extracted Morse-Smale complexes for each function sampled on a 512^2 grid using the steepest descent and our probabilistic link selection strategy, respectively. The right two columns show the result on a 4096^2 grid. Minima, saddles and maxima are shown as blue, yellow and red spheres, whereas the 0- and 1-separatrices are shown as blue and red lines, respectively. Since separatrices are integral lines of the gradient, the blue and red lines should follow the black lines. White arrows indicate regions where the difference of the two approaches is visually apparent.

their continuous counterpart. However, the presented idea is currently restricted to 2-dimensional cubical complexes.

To generalize the idea to general surfaces, a representation of the discrete metric provided by the surface is necessary. Only then a convergent behavior can be expected. Extending this idea to 3-dimensional image data is also not straight-forward. Combinatorial 1-streamlines can merge and split in three dimensions due to the fundamentally different structure of the cell graph (Section 3.2.4). It turns out that this property results in space-filling combinatorial separation surfaces if the links are chosen in the above probabilistic way. A more adapted link selection strategy is necessary.

6.2 The Importance of Separatrices

In Chapter 5, we introduced a novel importance measure to quantify the feature strength of separation lines and surfaces. As we have shown in several applications, this importance measure is able to reduce the topological structures to their essential part. The reduced structures allow for a meaningful analysis of the object under study.

The computation of separatrix persistence is based on two concepts: persistent homology and a multi-level representation of the Morse-Smale complex.

To compute the multi-level representation, pairs of critical points are sequentially removed. As discussed in Section 4.3, an optimal removal requires that the Morse-Smale complex is collapsible which is not always given in three dimensions. Because of this, the Morse-Smale complex cannot always be reduced to its minimal complexity based on the Morse inequalities. This is especially the case if a high level of noise is present.

Due to this property, separatrix persistence depends on the number of levels of detail in the hierarchy of combinatorial gradient fields \mathcal{V} . It may happen that only a small number of critical points can be removed from the complex. In this case, separatrix persistence captures only the local importance of the topological structures and fails to reflect their global importance. In our experiments, however, the topological complexity of the Morse-Smale complex could be sufficiently decreased such that separatrix persistence allowed for a meaningful reduction.

To address this limited view on the global scale, a topology-based smoothing of the scalar field could be beneficial. In case of noise-affected data, the 1-separation lines tend to create short split-merge sequences as discussed in Section 3.5.5, and parts of these sequences might not be resolved. However, such short split-merge sequences only describe local distortions in the data. A topological smoothing approach could remove these distortions while preserving the overall topological structure. Such a technique could be based on recently proposed methods such as [GZ07, WGS10, Bau11, JWS12]. However, a small-scale altering of the topological structures must be included. Given such a smoothed scalar field, one could re-run the topological analysis, and probably increase the view on the global scale.

Separatrix persistence assigns a uniquely well-defined importance measure to separation lines connecting the saddles with the extrema and to the separation surfaces. However, this is not the case for the separation lines that connect the two types of saddle in the 3-dimensional case. These lines are given as the intersection of the ascending and descending separation surfaces. Since each type of surfaces is assigned by separatrix persistence, its intersection is 2-valued at the 1-nodes. Depending on the point of view, the 1-separation lines have either a minimal or maximal character. The user needs to decide which characteristic is preferable in the analysis.

Topological methods as well as local approaches such as ridges/valleys extract image-based features. However, not all of them are meaningful in an application. For example, the aim of the application presented in Section 5.3.2, is to extract valleys of the Martian surface that describe erosional structures. However, not all creases are caused by these structures. A second example can be seen in the comparison Figure 5.11 for the salient edge extraction. The topological method tends to create connected extremal lines while ridge/valley definitions result in isolated lines. Which result is preferable, depends on the application specific task. The filtering due to separatrix persistence is purely image-based, no model knowledge is included. If the inclusion of such knowledge is wanted, specific filter criteria that depend on the application task need to be developed.

6.3 Extension to Higher Dimensions

The presented algorithmic framework works for 2- and 3-dimensional data given on a simplicial or cubical complex. One of the basic algorithm used within is the construction of the combinatorial gradient field. As shown by Robins et al. [RWS11], the critical points correspond 1-1 to the topological changes of the input data.

Theoretically, this algorithm is also applicable to higher dimensional cell complexes. However, the proof of Robins et al. [RWS11] is limited to three dimensions. While they assumed that their statements also hold in dimension four but using different arguments, it fails in dimensions greater than four. In practice, this results in falsely identified critical points, and therefore also in additional topological structures.

The Morse-Smale complex is defined by the intersection of the ascending and descending manifolds, and thereby dimensionless. The same holds for the restricted breadth-firsts search used in the algorithms presented in this thesis. Therefore, they can also be applied in higher dimensional complexes. However, an evaluation of their performance is open. The specific cell structure of higher dimensional complexes could increase the complexity of the presented algorithms, i.e., in which way combinatorial streamlines can merge and split. Additionally, the performances of the algorithms depend on the order of falsely identified critical points by Algorithm 5. Such points yield a multiple integration of a single topological structure.

6.4 From Discrete to Smooth

The combinatorial topological structures allow for an objective quantification of the input data. However, they can also be used to generate meaningful visualization enabling a deeper insight in the complex structure of the object under study.

Figure 6.8 shows a visualization of the electrostatic potential of the benzene molecule. It captures the essential parts of the field given by the separatrices, i.e., the minimal and maximal lines of the potential. The combinatorial 0- and 2-separation lines are computed as described in Chapter 3. Each point of these combinatorial separatrices serves as a seed point for a continuous stream line integration in forward and backward direction. Since integral lines in the vicinity of the separatrices are converging in forward or backward direction, this results in bundles of stream lines highlighting the flow behavior of the gradient of the electrostatic potential. The red stream lines emanate from the 2-separation lines, the blue stream lines from the 0-separation lines. The visualization elucidates the regular structure of the benzene molecule.

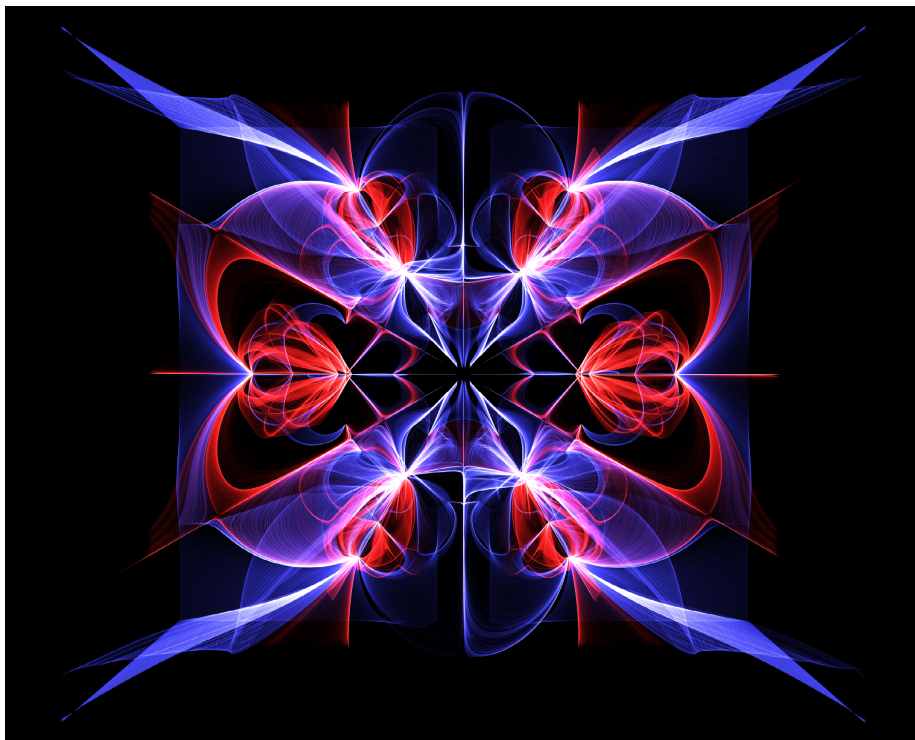


Figure 6.8: Visualization of the electrostatic potential around the benzene molecule.

Chapter 7

Conclusions and Future Work

In this thesis, we introduced a computational framework that allows for an efficient computation and quantification of topological structures in discrete scalar data. In particular, we presented a library of algorithms allowing to efficiently compute the critical points, combinatorial separation lines and surfaces. Since noise and sampling artifacts may create additional structures, we also presented algorithms to create a multi-level representation of these structures and introduced the concept of separatrix persistence to assess the feature strength of separation lines and surfaces.

The presented algorithms are purely combinatorial based on the discrete Morse theory of Robin Forman [For98b]. In contrast to previous approaches, we presented an algorithm that computes the Morse-Smale complex with a provably optimal complexity of $O(cn) \subset O(n^2)$. We also presented algorithms that compute a multi-level representation of this complex for the 2- and 3-dimensional case. The combinatorial setting increases the robustness of our algorithms and guarantees the consistency of their output. In contrast to previous work, the relative memory consumption is constant *and* the running time behaves almost linear for well-defined data.

All the algorithms presented in the course of this thesis do not depend on any computational parameter. The computed Morse-Smale complex, its multi-level representation, and the quantification of the topological structures solely depend on the information of the input data. This enables an objective analysis. The user only chooses an appropriate level of detail for further analysis.

As we have shown, this complex can also be efficiently computed for large 3-dimensional scalar data ($> 1024^3$) on commodity hardware. It would be interesting to investigate an out-of-core realization of the presented framework since this would enable its applicability to very large data. A first approach was proposed by Gyulassy et al. [GBHP08] and some of the presented ideas could probably also be used in our setting.

The Morse-Smale complex describes the input data completely from the homological point of view. We presented a strategy that allows to compute persistent homology using this complex with an overall output-sensitive complexity of $O(cn + c^3)$. In contrast to the commonly used algebraic techniques, this strategy requires significantly less memory. For a data set of a size of about 4 GB, our strategy reduces the memory consumption from 500 GB to 14 GB.

A fundamental question, which is still an open problem in the literature, is the relation between the topological complexity of a given input data and the persistence computation times. Since matrix reduction is a global operation, the complexity of the

underlying Morse-Smale complex is crucial.

We used persistence to compute meaningful features of two near-isometric surfaces. We presented a conceptually direct and simple algorithmic pipeline that is able to compute accurately a correspondence between the features of the two surfaces.

We introduced a novel importance measure called *separatrix persistence* that allows for a meaningful filtering of separation lines and surfaces. Since this measure enjoys the global nature of the multi-level representation of the Morse-Smale complex and is derived from persistent homology, it behaves stable with respect to noise. We have evaluated our new measure on a variety of data sets from different domains, which illustrates its robustness and applicability.

We thoroughly compared our new importance measure to local approaches such as ridges/valleys. It is very interesting that certainly both concepts can give very similar results. This is especially the case for the commonly used definition of Height Ridges. As discussed by Schindler et al. [SPFT12], this definition states that ridge lines cannot be integral lines, in contrast to separatrices. Those lines are by definition integral lines. However, the computed extremal structures largely coincide in our experiments. In this thesis, we discussed the relationship between our topological approach and Height Ridges. It would be beneficial to further investigate the similarities of these concepts to bring new insights in the area of feature extraction.

Besides the shown application cases, this framework could be extended to the combinatorial analysis of 3-dimensional time-dependent scalar data. First promising results were already presented by Reininghaus et al. [RKWH12] and Kasten et al. [KRHH11] for 2-dimensional data. While the tracking of the minima and maxima can be done in a similar fashion, an extension needs to target the challenging question how 1- and 2-saddles can be robustly tracked over time. Such techniques are of great interest in the area of turbulence research in physics [LBM⁺06].

From the application point of view, the convergence of separation lines and surfaces is still an open question. As discussed in Chapter 6, a probabilistic approach yields convergent separation lines in two dimensions. While this observation is only of empirical nature, a formal proof of convergence would substantiate the experiments. The same approach is not directly applicable in three dimensions, since it might result in space filling surfaces due to the specific structure of the cell complex. The multi-level representation presented in this thesis uses the intersection of surfaces, and, in this case, its computation is therefore very challenging using the probabilistic approach in three dimensions.

The reconstruction of a smooth scalar field based on the information provided by the topological structures is an interesting application of this framework. The Morse-Smale complex induces a decomposition of the data into monotone regions. Its multi-level representation merges implicitly similar regions based on the evolution of iso-levels. This information could be used to create a multi-scale representation of the scalar data itself. For the 2-dimensional case, efficient algorithms were already proposed [WGS10, JWS12]. An extension to 3-dimensional data, however, is not yet done.

Bibliography

- [AC84] J. Aubin and A. Cellina. *Differential Inclusions: Set-Valued Maps and Viability Theory*. Grundlehren Der Mathematischen Wissenschaften. Springer-Verlag, 1984.
- [Aim] Aim@Shape. <http://shapes.aim-at-shape.net/>.
- [Ban70] T. F. Banchoff. Critical points and curvature for embedded polyhedral surfaces. *The American Mathematical Monthly*, 77(5):pp. 475–485, 1970.
- [Bar] D. Bartz. Volvis: Voxel data repository. <http://www.volvis.org/>.
- [Bau11] U. Bauer. *Persistence in discrete Morse Theory*. PhD thesis, University of Göttingen, 2011.
- [BBC⁺10] A. Bronstein, M. Bronstein, U. Castellani, A. Dubrovina, L. Guibas, R. P. Horaud, R. Kimmel, D. Knossow, E. von Lavante, D. Mateus, M. Ovsjanikov, and A. Sharma. Shrec 2010: Robust correspondence benchmark. In *Eurographics Workshop on 3D Object Retrieval*, 2010.
- [BBK08] A. Bronstein, M. Bronstein, and R. Kimmel. *Numerical Geometry of Non-Rigid Shapes*. Springer Publishing Company, Incorporated, 1 edition, 2008.
- [BCM05] A. Buades, T. Coll, and J.-M. Morel. A review of image denoising algorithms, with a new one. *Multiscale Modeling and Simulation*, 4:490–530, 2005.
- [BEHP04] P.-T. Bremer, H. Edelsbrunner, B. Hamann, and V. Pascucci. A topological hierarchy for functions on triangulated surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 10:385–396, 2004.
- [BEK10] P. Bendich, H. Edelsbrunner, and M. Kerber. Computing robustness and persistence for images. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1251–1260, 2010.
- [Ber13] J. Bernoulli. *Ars conjectandi*. impensis Thurnisiorum, fratrum, 1713.
- [BGK10] Z. I. Botev, J. F. Grotowski, and D. P. Kroese. Kernel density estimation via diffusion. *Annals of Statistics*, 38(5):2916–2957, 2010.
- [Ble10] J. Bleacher. NASA press release: Lava likely made river-like channel on mars, 2010. <http://www.nasa.gov/topics/solarsystem/features/mars-lava-channels.html>.

- [BLW12] U. Bauer, C. Lange, and M. Wardetzky. Optimal topological simplification of discrete functions on surfaces. *Discrete and Computational Geometry*, 47:347–377, 2012.
- [Bot88] R. Bott. Morse theory indomitable. In *Publications Mathématiques*, 68, pages 99–114. Institut des Hautes Études Scientifiques, 1988.
- [Car95] M. H. Carr. The martian drainage system and the origin of valley networks and fretted channels. *Journal of Geophysical Research*, 100(E4):7479–7507, 1995.
- [Cay59] A. Cayley. On contour and slope lines. *The London, Edinburg and Dublin Philosophical Magazine and Journal of Science*, 18:264–268, 1859.
- [CCL03] F. Cazals, F. Chazal, and T. Lewiner. Molecular shape analysis based upon the Morse-Smale complex and the Connolly function. In *SCG '03: Proceedings of the nineteenth annual symposium on Computational geometry*, pages 351–360, New York, NY, USA, 2003. ACM Press.
- [CDS⁺12] G. Chen, Q. Deng, A. Szymczak, R. S. Laramée, and E. Zhang. Morse set classification and hierarchical refinement using conley index. *IEEE Transactions on Visualization and Computer Graphics*, 18:767–782, 2012.
- [CFL11] C. Chen, D. Freedman, and C. H. Lampert. Enforcing topological constraints in random field image segmentation. In *CVPR'11*, pages 2089–2096, 2011.
- [Cha00] M. K. Chari. On discrete morse functions and combinatorial decompositions. *Discrete Mathematics*, 217(1-3):101–113, 2000.
- [Cha11] A. Chattopadhyay. *Certified Geometric Computation: Radial Basis Function based Isosurfaces and Morse-Smale Complexes*. PhD thesis, University of Groningen, 2011.
- [CK11a] C. Chen and M. Kerber. An Output-Sensitive Algorithm for Persistent Homology. In *Proceedings of the 27th annual symposium on Computational geometry*, 2011.
- [CK11b] C. Chen and M. Kerber. Persistent homology computation with a twist. In *27th European Workshop on Comp. Geometry (EuroCG 2011)*, 2011.
- [CMEH⁺04] K. Cole-McLaughlin, H. Edelsbrunner, J. Harer, V. Natarajan, and V. Pascucci. Loops in reeb graphs of 2-manifolds. *Discrete Comput. Geom.*, 32(2):231–244, July 2004.
- [CMLZ08] G. Chen, K. Mischaikow, R. S. Laramée, and E. Zhang. Efficient Morse decompositions of vector fields. *IEEE Transactions on Visualization and Computer Graphics*, 14(4):848–862, July 2008.
- [Coh73] M. Cohen. *A course in simple-homotopy theory*. Springer-Verlag, 1973.

- [Con78] C. Conley. *Isolated Invariant Sets and the Morse Index*. Number no. 38 in Regional Conference Series in Mathematics. Conference Board of the Mathematical Sciences, 1978.
- [CP03] F. Cazals and M. Pouget. Estimating differential quantities using polynomial fitting of osculating jets. In *Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, SGP '03, pages 177–187, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.
- [CSA00] H. Carr, J. Snoeyink, and U. Axen. Computing contour trees in all dimensions. In *SODA '00: Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms*, pages 918–926, Philadelphia, PA, USA, 2000. Society for Industrial and Applied Mathematics.
- [CSEH07] D. Cohen-Steiner, H. Edelsbrunner, and J. Harer. Stability of persistence diagrams. *Discrete and Computational Geometry*, 37:103–120, 2007.
- [CSEH09] D. Cohen-Steiner, H. Edelsbrunner, and J. Harer. Extending persistence using poincaré and lefschetz duality. *Found. Comput. Math.*, 9(1):79–103, Jan. 2009.
- [CSvdP04] H. Carr, J. Snoeyink, and M. van de Panne. Simplifying flexible isosurfaces using local geometric measures. In *Proceedings of the conference on Visualization '04*, VIS '04, pages 497–504, Washington, DC, USA, 2004. IEEE Computer Society.
- [CVY12] A. Chattopadhyay, G. Vegter, and C. Yap. Certified computation of planar Morse-Smale complexes of smooth functions. In *Proceedings of the 28th ACM Symposium on Computational Geometry (SoCG 2012)*, pages 1–12, 2012.
- [Dam99] J. Damon. Properties of ridges and cores for two-dimensional images. *Journal of Mathematical Imaging and Vision*, 10:163–174, 1999.
- [DBG⁺06] S. Dong, P.-T. Bremer, M. Garland, V. Pascucci, and J. C. Hart. Spectral surface quadrangulation. *ACM Transaction on Graphics*, 25(3):1057–1066, July 2006.
- [DFR⁺] D. DeCarlo, A. Finkelstein, S. Rusinkiewicz, A. Santella, M. Burns, and J. Klawe. Suggestive contours software. <http://www.cs.princeton.edu/gfx/proj/sugcon/>.
- [DLL⁺10] T. Dey, K. Li, C. Luo, P. Ranjan, I. Safa, and Y. Wang. Persistent heat signature for pose-oblivious matching of incomplete models. *Computer Graphics Forum*, 29(5):1545–1554, 2010.
- [dLvL99a] W. de Leeuw and R. van Liere. Collapsing flow topology using area metrics. In *Proc. IEEE Visualization '99*, pages 149–354, 1999.
- [dLvL99b] W. de Leeuw and R. van Liere. Visualization of global flow structures using multiple levels of topology. In *Data Visualization 1999. Proc. VisSym 99*, pages 45–52, 1999.

- [dSV52] B. de Saint-Venant. Surfaces à plus grande pente constituées sur les lignes courbes. *Bulletin de la soc. philomath. de Paris*, 1852.
- [Ebe96] D. Eberly. *Ridges in Image and Data Analysis*. Kluwer Academic Publishers, 1996.
- [EH08] H. Edelsbrunner and J. Harer. Persistent homology — a survey. In J. E. Goodman, J. Pach, and R. Pollack, editors, *Surveys on Discrete and Computational Geometry: Twenty Years Later*, volume 458, pages 257–282. AMS Bookstore, 2008.
- [EH10] H. Edelsbrunner and J. Harer. *Computational Topology. An Introduction*. American Mathematical Society, January 2010.
- [EHNP03] H. Edelsbrunner, J. Harer, V. Natarajan, and V. Pascucci. Morse-smale complexes for piecewise linear 3-manifolds. In *SCG '03: Proceedings of the nineteenth annual symposium on Computational geometry*, pages 361–370, New York, NY, USA, 2003. ACM.
- [EHZ03] H. Edelsbrunner, J. Harer, and A. Zomorodian. Hierarchical Morse complexes for piecewise linear 2-manifolds. *Discrete Computational Geometry*, 30:87–107, 2003.
- [ELZ02] H. Edelsbrunner, D. Letscher, and A. Zomorodian. Topological persistence and simplification. *Discrete Computational Geometry*, 28:511–533, 2002.
- [FAT99] I. Fujishiro, T. Azuma, and Y. Takeshima. Automating transfer function design for comprehensible volume rendering based on 3d field topology analysis (case study). In *Proceedings of the conference on Visualization '99: celebrating ten years, VIS '99*, pages 467–470, Los Alamitos, CA, USA, 1999. IEEE Computer Society Press.
- [FL99] P. Frosini and C. Landi. Size theory as a topological tool for computer vision. *Pattern Recognition And Image Analysis*, 9(4):596–603, 1999.
- [For98a] R. Forman. Combinatorial vector fields and dynamical systems. *Mathematische Zeitschrift*, 228:629–681, 1998.
- [For98b] R. Forman. Morse theory for cell complexes. *Advances in Mathematics*, 134:90–145, 1998.
- [For01] R. Forman. A user's guide to discrete morse theory. In *Proceedings of the 2001 International Conference on Formal Power Series and Algebraic Combinatorics*, Advances in Applied Mathematics, 2001.
- [FP01] J. D. Furst and S. M. Pizer. Marching ridges. In M. H. Hamza, editor, *SIP*, pages 22–26. IASTED/ACTA Press, 2001.
- [Fra06] J. Frank, editor. *Electron Tomography*. Springer, 2006.
- [Gau28] C. Gauss. *Disquisitiones generales circa superficies curvas*. Typis Dieterichianis, 1828.

- [GBHP08] A. Gyulassy, P.-T. Bremer, B. Hamann, and V. Pascucci. A practical approach to Morse-Smale complex computation: scalability and generality. *IEEE Transactions on Visualization and Computer Graphics*, 14:1619–1626, 2008.
- [GBPH11] A. Gyulassy, P.-T. Bremer, V. Pascucci, and B. Hamann. Practical considerations in Morse-Smale complex computation. In V. Pascucci, X. Tricoche, H. Hagen, and J. Tierny, editors, *Topological Methods in Data Analysis and Visualization*, Mathematics and Visualization, pages 67–78. Springer Berlin Heidelberg, 2011.
- [GC95] L. D. Griffin and A. C. Colchester. Superficial and deep structure in linear diffusion scale space: isophotes, critical points and separatrices. *Image & Vision Comp.*, 13(7):543 – 557, 1995.
- [GGK] M. E. Gröller, G. Glaeser, and J. Kastner.
<http://www.cg.tuwien.ac.at/research/publications/2005/dataset-stagbeetle/>. Stagbeetle.
- [GMW⁺10] D. Günther, P. C. McGuire, S. Walter, T. Weinkauff, and H.-C. Hege. Extraction of valley networks in Mars elevation maps. In *European Planetary Science Congress 2010*, page 216, Sept. 2010.
- [GNP⁺06] A. Gyulassy, V. Natarajan, V. Pascucci, P.-T. Bremer, and B. Hamann. A topological approach to simplification of three-dimensional scalar functions. *IEEE Transactions on Visualization and Computer Graphics*, 12(4):474–484, 2006.
- [GNPH07] A. Gyulassy, V. Natarajan, V. Pascucci, and B. Hamann. Efficient computation of Morse-Smale complexes for three-dimensional scalar functions. *IEEE Transactions on Visualization and Computer Graphics*, 13:1440–1447, 2007.
- [GRP⁺11] D. Günther, J. Reininghaus, S. Prohaska, T. Weinkauff, and H.-C. Hege. Efficient computation of a hierarchy of discrete 3D gradient vector fields. In *Proceedings TopoInVis*. Springer, Zürich, Switzerland, April 2011.
- [GRWH11] D. Günther, J. Reininghaus, H. Wagner, and I. Hotz. Memory efficient computation of persistent homology for 3D image data using discrete Morse theory. In T. Lewiner and R. Torres, editors, *Proceedings...*, pages 25–32, Maceió, Los Alamitos, august 2011. Conference on Graphics, Patterns and Images, 24. (SIBGRAPI), IEEE.
- [GRWH12] D. Günther, J. Reininghaus, H. Wagner, and I. Hotz. Efficient computation of 3D Morse-Smale complexes and persistent homology using discrete Morse theory. *The Visual Computer*, 28:959–969, 2012.
- [GSW12] D. Günther, H.-P. Seidel, and T. Weinkauff. Extraction of dominant extremal structures in volumetric data using separatrix persistence. *Computer Graphics Forum*, 2012. to appear.

- [Gyu08] A. Gyulassy. *Combinatorial Construction of Morse-Smale Complexes for Data Analysis and Visualization*. PhD thesis, University of California, Davis, 2008.
- [GZ07] Y. I. Gingold and D. Zorin. Controlled-topology filtering. *Computer-Aided Design*, 39(8):676–684, 2007.
- [Hal01] G. Haller. Distinguished material surfaces and coherent structures in three-dimensional fluid flows. *Physica D*, 149:248–277, 2001.
- [Hat02] A. Hatcher. *Algebraic Topology*. Cambridge University Press, Cambridge, U.K., 2002.
- [Hau14] F. Hausdorff. *Grundzüge der Mengenlehre*. Chelsea Publishing Series. Chelsea Publishing Company, 1914.
- [HAWG08] Q.-X. Huang, B. Adams, M. Wicke, and L. J. Guibas. Non-rigid registration under isometric deformations. In *Proceedings of the Symposium on Geometry Processing*, SGP '08, pages 1449–1457, Aire-la-Ville, Switzerland, Switzerland, 2008. Eurographics Association.
- [HP03] B. M. Hynek and R. J. Phillips. New data reveal mature, integrated drainage systems on mars indicative of past precipitation. *Geology*, 31(9):757–760, Sep. 2003.
- [HPW05] K. Hildebrandt, K. Polthier, and M. Wardetzky. Smooth feature lines on surface meshes. In *Proceedings of the third Eurographics symposium on Geometry processing*, SGP '05, Aire-la-Ville, Switzerland, Switzerland, 2005. Eurographics Association.
- [Hun87] J. C. R. Hunt. Vorticity and vortex dynamics in complex turbulent flows. *CANCAM, Transactions Canadian Society for Mechanical Engineering*, 11(1):21–35, 1987.
- [JP06] M. Joswig and M. E. Pfetsch. Computing optimal Morse matchings. *SIAM Journal on Discrete Mathematics*, 20(1):11–25, 2006.
- [JWS12] A. Jacobson, T. Weinkauff, and O. Sorkine. Smooth shape-aware functions with controlled extrema. *Computer Graphics Forum (proceedings of EUROGRAPHICS/ACM SIGGRAPH Symposium on Geometry Processing)*, 31(5):1577–1586, 2012.
- [Kar05] B. Karlsson. *Beyond the C++ Standard Library: An Introduction to Boost*. Pearson Education, 2005.
- [KHNH12] J. Kasten, I. Hotz, B. Noack, and H.-C. Hege. Vortex merge graphs in two-dimensional unsteady flow fields. In *Proceedings Joint EG - IEEE TCVG Symposium on Visualization 2012 – Short Paper*, 2012.
- [Kim] V. Kim. Surface correspondence. <http://www.cs.princeton.edu/vk/CorrsCode/>.
- [KKH12] N. Kotava, A. Knoll, and H. Hagen. Morse-Smale decomposition of multivariate transfer function space for separably-sampled volume rendering. *Computer Aided Geometric Design*, (0):–, 2012. (to appear).

- [KKM05] H. King, K. Knudson, and N. Mramor. Generating discrete Morse functions from point data. *Experimental Mathematics*, 14(4):435–444, 2005.
- [KLF11] V. G. Kim, Y. Lipman, and T. Funkhouser. Blended intrinsic maps. *ACM Transactions on Graphics*, 30(4):79:1–79:12, July 2011.
- [KMM04] T. Kaczynski, K. Mischaikow, and M. Mrozek. *Computational Homology*, volume 157 of *Applied Mathematical Sciences*. Springer-Verlag, 2004.
- [Kov01] V. Kovalevsky. Algorithms and data structures for computer topology. *Lecture Notes in Computer Science*, 2243:38–58, 2001.
- [KRHH11] J. Kasten, J. Reininghaus, I. Hotz, and H.-C. Hege. Two-dimensional time-dependent vortex regions based on the acceleration magnitude. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2080–2087, Dec. 2011.
- [Kuh55] H. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97, 1955.
- [KvD93] J. J. Koenderinck and A. van Doorn. Local features of smooth shape: Ridges and courses. *SPIE Proc. Geometric Methods in Computer Vision II*, 2031(2031):2–13, 1993.
- [LBM⁺06] D. Laney, P. T. Bremer, A. Mascarenhas, P. Miller, and V. Pascucci. Understanding the structure of the turbulent mixing layer in hydrodynamic instabilities. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1053–1060, 2006.
- [Lew05] T. Lewiner. *Geometric discrete Morse complexes*. PhD thesis, Department of Mathematics, PUC-Rio, 2005. Advised by Hélio Lopes and Geovan Tavares.
- [Lew12] T. Lewiner. Critical sets in discrete Morse theories: Relating Forman and piecewise-linear approaches. *Computer Aided Geometric Design*, (0):–, 2012. (to appear).
- [LF09] Y. Lipman and T. Funkhouser. Möbius voting for surface correspondence. *ACM Transactions on Graphics*, 28(3):72:1–72:12, July 2009.
- [LH05] M. Leordeanu and M. Hebert. A spectral technique for correspondence problems using pairwise constraints. In *Proceedings of the Tenth IEEE International Conference on Computer Vision - Volume 2, ICCV '05*, pages 1482–1489, Washington, DC, USA, 2005. IEEE Computer Society.
- [LHZIP07] R. S. Laramée, H. Hauser, L. Zhao, and F. H. Post. Topology-based flow visualization, the state of the art. In H. H. Helwig Hauser and H. Theisel, editors, *Topology-based Methods in Visualization*, Mathematics and Visualization, pages 1–19. Springer Berlin Heidelberg, May 2007.

- [LLSV99] A. Lopez, F. Lumbreras, J. Serrat, and J. Villanueva. Evaluation of methods for ridge and valley detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 21(4):327–335, apr 1999.
- [LLT03] T. Lewiner, H. Lopes, and G. Tavares. Optimal discrete morse functions for 2-manifolds. *Computational Geometry*, 26(3):221–233, 2003.
- [LLT04] T. Lewiner, H. Lopes, and G. Tavares. Applications of forman’s discrete morse theory to topology visualization and mesh compression. *IEEE Transactions on Visualization and Computer Graphics*, 10(5):499–508, 2004.
- [LS09] W. Luo and T. F. Stepinski. Computer-generated global map of valley networks on Mars. *Journal of Geophysical Research*, 114(E11010):11, 2009.
- [Max70] J. C. Maxwell. On hills and dales. *The London, Edinburg and Dublin Philosophical Magazine and Journal of Science*, 40:421–425, 1870.
- [MERT12] J. Martinez Esturo, C. Rössl, and H. Theisel. Continuous deformations by isometry preserving shape integration. In *Proceedings of the 7th international conference on Curves and Surfaces*, pages 456–472, Berlin, Heidelberg, 2012. Springer-Verlag.
- [Mil63] J. Milnor. *Morse Theory*. Princeton University Press, 1963.
- [Mil65a] J. Milnor. *Lectures on the H-Cobordism Theorem*. Mathematical notes. Princeton University Press, 1965.
- [Mil65b] J. Milnor. *Topology from the differentiable viewpoint*. Univ. Press Virginia, 1965.
- [MMS11] N. Milosavljevic, D. Morozov, and P. Skraba. Zigzag Persistent Homology in Matrix Multiplication Time. In *Proceedings of the 27th annual symposium on Computational geometry*, 2011.
- [Mor34] M. Morse. *The Calculus of Variations in the Large*. Number v. 18 in Colloquium Publications - American Mathematical Society. American Mathematical Society, 1934.
- [Mor66] S. P. Morse. Topological approach to obtain ground track of aircraft using height over terrain and contour map. Technical report, NASA, Jan. 1966.
- [Mor05] D. Morozov. Persistence algorithm takes cubic time in the worst case. In *BioGeometry News*. Duke Computer Science, Durham, NC, 2005.
- [MS95] D. R. Musser and A. Saini. *The STL Tutorial and Reference Guide: C++ Programming with the Standard Template Library*. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 1995.
- [MW10] M. Mrozek and T. Wanner. Coreduction homology algorithm for inclusions and persistent homology. *Computers and Mathematics with Applications*, 60(10):2812–2833, Nov. 2010.

- [New69] I. Newton. De analysi per aequationes numero terminorum infinitas. *Commercium Epistolicum D. Johannis Collins et aliorum de analysi promota*, pages 3–20, 1669. published in 1711.
- [NS94] L. Najman and M. Schmitt. Watershed of a continuous function. *Signal Processing*, 38(1):99 – 112, 1994.
- [OMMG10] M. Ovsjanikov, Q. Mérigot, F. Mémoli, and L. J. Guibas. One point isometric matching with the heat kernel. *Computer Graphics Forum*, 29(5):1555–1564, 2010.
- [PGW12] R. Peikert, D. Günther, and T. Weinkauff. Comment on "second derivative ridges are straight lines and the implications for computing lagrangian coherent structures, physica d 2012.05.006". *Physica D*, 241(accepted), to appear 2012.
- [POS⁺11] C. Pagot, D. Osmari, F. Sadlo, D. Weiskopf, T. Ertl, and J. Comba. Efficient parallel vectors feature extraction from higher-order data. *Computer Graphics Forum*, 30(3):751–760, 2011.
- [PR99] R. Peikert and M. Roth. The "parallel vectors" operator - a vector field visualization primitive. In *Proceedings of the 10th IEEE Visualization 1999 Conference (VIS '99)*, Visualization '99, pages 263–270, Washington, DC, USA, 1999. IEEE Computer Society.
- [PS08] R. Peikert and F. Sadlo. Height Ridge Computation and Filtering for Visualization. In I. Fujishiro, H. Li, and K.-L. Ma, editors, *Proceedings of Pacific Vis 2008*, pages 119–126, March 2008.
- [Rap90] J. Raphson. *Analysis Aequationum universalis*. 1690.
- [Ree46] G. Reeb. Sur les points singuliers d'une forme de Pfaff complètement intégrable ou d'une fonction numérique. *Comptes Rendus de L'Académie ses Sciences*, pages 847–849, 1946.
- [RGH⁺10] J. Reininghaus, D. Günther, I. Hotz, S. Prohaska, and H.-C. Hege. TADD: A computational framework for data analysis using discrete Morse theory. In K. Fukuda, J. van der Hoeven, M. Joswig, and N. Takayama, editors, *Mathematical Software – ICMS 2010*, volume 6327 of *Lecture Notes in Computer Science*, pages 198–208. Springer, 2010.
- [RGH⁺12a] J. Reininhaus, D. Günther, I. Hotz, T. Weinkauff, and H.-P. Seidel. Combinatorial gradient fields for 2D images with empirically convergent separatrices, 2012. Arxiv:1208.6523v1.
- [RGH⁺12b] A. Rigort, D. Günther, R. Hegerl, D. Baum, B. Weber, S. Prohaska, O. Medalia, W. Baumeister, and H.-C. Hege. Automated segmentation of electron tomograms for a quantitative description of actin filament networks. *Journal of Structural Biology*, 177(1):135–144, 2012.
- [RKG⁺11] J. Reininghaus, N. Kotava, D. Günther, J. Kasten, H. Hagen, and I. Hotz. A scale space based persistence measure for critical points in 2d scalar fields. *IEEE Transactions on Visualization and Computer Graphics*, 17:2045–2052, Dec. 2011.

- [RKWH12] J. Reininghaus, J. Kasten, T. Weinkauff, and I. Hotz. Efficient computation of combinatorial feature flow fields. *IEEE Transactions on Visualization and Computer Graphics*, 18(9):1563 – 1573, 2012.
- [Rob00] V. Robins. *Computational topology at multiple resolutions: foundations and applications to fractals and dynamics*. PhD thesis, University of Colorado at Boulder, Boulder, CO, USA, 2000.
- [Röt] S. Röttger. The Volume Library. Computer Graphics Group, University of Erlangen, <http://www9.informatik.uni-erlangen.de/External/vollib/>.
- [RPP⁺09] O. Rosanwo, C. Petz, S. Prohaska, I. Hotz, and H.-C. Hege. Dual streamline seeding. In P. Eades, T. Ertl, and H.-W. Shen, editors, *Proceedings of the IEEE Pacific Visualization Symposium*, pages 9 – 16, 2009.
- [RWS11] V. Robins, P. J. Wood, and A. P. Sheppard. Theory and algorithms for constructing discrete morse complexes from grayscale digital images. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 33(8):1646–1658, 2011.
- [Sch93] M. Schwarz. *Morse homology*. Progress in Mathematics. Birkhäuser, 1993.
- [SHCB11] A. Sharma, R. Horaud, J. Cech, and E. Boyer. Topologically-robust 3D shape matching based on diffusion geometry and seed growing. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2481–2488, 2011.
- [Sma61a] S. Smale. Generalized poincare’s conjecture in dimensions greater than four. *The Annals of Mathematics*, 74(2):391–406, Sep. 1961.
- [Sma61b] S. Smale. On gradient dynamical systems. *The Annals of Mathematics*, 74:199–206, 1961.
- [SN12] N. Shivashankar and V. Natarajan. Parallel computation of 3D Morse-Smale complexes. *Computer Graphics Forum*, 31(3pt1):965–974, 2012.
- [SOG09] J. Sun, M. Ovsjanikov, and L. Guibas. A concise and provably informative multi-scale signature based on heat diffusion. In *Proceedings of the Symposium on Geometry Processing*, SGP ’09, pages 1383–1392, Aire-la-Ville, Switzerland, Switzerland, 2009. Eurographics Association.
- [SPFT12] B. Schindler, R. Peikert, R. Fuchs, and H. Theisel. Ridge concepts for the visualization of Lagrangian coherent structures. In *Topological Methods in Data Analysis and Visualization II*, pages 221–235. Springer, 2012.
- [SS96] D. Stalling and T. Steinke. Visualization of vector fields in quantum chemistry. ZIB Preprint SC-96-01, 1996.
- [SSK⁺05] V. Surazhsky, T. Surazhsky, D. Kirsanov, S. J. Gortler, and H. Hoppe. Fast exact and approximate geodesics on meshes. *ACM Transactions on Graphics*, 24(3):553–560, July 2005.

- [SWTH07] J. Sahner, T. Weinkauff, N. Teuber, and H.-C. Hege. Vortex and strain skeletons in eulerian and lagrangian frames. *IEEE Transactions on Visualization and Computer Graphics*, 13(5):980–990, September - October 2007.
- [SZ12] A. Szymczak and E. Zhang. Robust morse decompositions of piecewise constant vector fields. *IEEE Transactions on Visualization and Computer Graphics*, 18(6):938–951, June 2012.
- [TBW⁺09] A. Tevs, M. Bokeloh, M. Wand, A. Schilling, and H.-P. Seidel. Isometric registration of ambiguous and partial data. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1185–1192, june 2009.
- [TGSP09] J. Tierny, A. Gyulassy, E. Simon, and V. Pascucci. Loop surgery for volumetric meshes: Reeb graphs reduced to contour trees. *IEEE Transactions on Visualization and Computer Graphics*, 15:1177–1184, 2009.
- [Thi96] J.-P. Thirion. The extremal mesh and the understanding of 3D surfaces. *International Journal of Computer Vision*, 19(2):115 – 128, 1996.
- [TM10] T. Tung and T. Matsuyama. Dynamic surface matching by geodesic mapping for 3D animation transfer. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1402–1409, june 2010.
- [TRS03a] H. Theisel, C. Rössl, and H.-P. Seidel. Combining topological simplification and topology preserving compression for 2d vector fields. In *Proc. Pacific Graphics 2003*, pages 419–423, 2003.
- [TRS03b] H. Theisel, C. Rössl, and H.-P. Seidel. Compression of 2D vector fields under guaranteed topology preservation. *Computer Graphics Forum (Eurographics 2003)*, 22(3):333–342, 2003.
- [TS03] H. Theisel and H.-P. Seidel. Feature flow fields. In *VisSym '03: Proceedings of the symposium on Data Visualization 2003*, pages 141–148, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.
- [TSH00] X. Tricoche, G. Scheuermann, and H. Hagen. A topology simplification method for 2D vector fields. In *Proc. IEEE Visualization 2000*, pages 359–366, 2000.
- [TV98] S. P. Tarasov and M. N. Vyalyi. Construction of contour trees in 3d in $o(n \log n)$ steps. In *Proceedings of the fourteenth annual symposium on Computational geometry, SCG '98*, pages 68–75, New York, NY, USA, 1998. ACM.
- [TWHS03a] H. Theisel, T. Weinkauff, H.-C. Hege, and H.-P. Seidel. Saddle connectors - an approach to visualizing the topological skeleton of complex 3d vector fields. In G. Turk, J. J. van Wijk, and R. Moorhead, editors, *Proc. IEEE Visualization 2003*, pages 225–232, Seattle, U.S.A., October 2003.

- [TWHS03b] H. Theisel, T. Weinkauff, H.-C. Hege, and H.-P. Seidel. Saddle connectors - an approach to visualizing the topological skeleton of complex 3D vector fields. In *Proceedings IEEE Visualization 2003*, pages 225–232, 2003.
- [TWHS07] H. Theisel, T. Weinkauff, H.-C. Hege, and H.-P. Seidel. On the applicability of topological methods for complex flow data. In H. Hauser, H. Hagen, and H. Theisel, editors, *Topology-based Methods in Visualization*, Mathematics and Visualization, pages 105–120. Springer, 2007. Topo-In-Vis 2005, Budmerice, Slovakia, September 29 - 30.
- [vKZHCO11] O. van Kaick, H. Zhang, G. Hamarneh, and D. Cohen-Or. A survey on shape correspondence. *Computer Graphics Forum*, 30(6):1681–1707, 2011.
- [VS91] L. Vincent and P. Soille. Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6):583–598, June 1991.
- [WBP07] G. Weber, P.-T. Bremer, and V. Pascucci. Topological landscapes: A terrain metaphor for scientific data. *Visualization and Computer Graphics, IEEE Transactions on*, 13(6):1416–1423, nov.-dec. 2007.
- [WCV12] H. Wagner, C. Chen, and E. Vucini. Efficient computation of persistent homology for cubical data. In R. Peikert, H. Hauser, H. Carr, and R. Fuchs, editors, *Topological Methods in Data Analysis and Visualization II*, Mathematics and Visualization, pages 91–108. Springer, 2012. TopoInVis 2011, Zürich, Switzerland, April 4 - 6.
- [WDC⁺07] G. H. Weber, S. E. Dillard, H. Carr, V. Pascucci, and B. Hamann. Topology-controlled volume rendering. *IEEE Trans. Vis. Comput. Graph.*, 13(2):330–341, 2007.
- [Wei08] T. Weinkauff. *Extraction of Topological Structures in 2D and 3D Vector Fields*. PhD thesis, University Magdeburg, 2008.
- [WG09] T. Weinkauff and D. Günther. Separatrix persistence: Extraction of salient edges on surfaces using topological methods. *Computer Graphics Forum (Proc. SGP '09)*, 28(5):1519–1528, July 2009.
- [WGS10] T. Weinkauff, Y. Gingold, and O. Sorkine. Topology-based smoothing of 2D scalar fields with c1-continuity. *Computer Graphics Forum (Proc. EuroVis)*, 29(3):1221–1230, June 2010.
- [Whi49] J. H. C. Whitehead. Combinatorial homotopy. I. *Bulletin of the American mathematical Society*, 55(5):213–245, 1949.
- [WS02] G. H. Weber and G. Scheuermann. Topology-based transfer function design. In J. J. Villanueva, editor, *Proceedings of the Second IASTED International Conference on Visualization, Imaging, and Image Processing*, pages 527–532, Anaheim, CA, USA,, 2002. ACTA Press.
- [WTGP11] T. Weinkauff, H. Theisel, A. V. Gelder, and A. Pang. Stable Feature Flow Fields. *IEEE Transactions on Visualization and Computer Graphics*, 17(6):770–780, June 2011.

- [WTS⁺05] T. Weinkauff, H. Theisel, K. Shi, H.-C. Hege, and H.-P. Seidel. Extracting higher order critical points and topological simplification of 3D vector fields. In *Proc. IEEE Visualization 2005*, pages 559–566, Minneapolis, U.S.A., October 2005.
- [YBS05] S. Yoshizawa, A. Belyaev, and H.-P. Seidel. Fast and robust detection of crest lines on meshes. In *Proceedings of the 2005 ACM symposium on Solid and physical modeling*, SPM '05, pages 227–232, New York, NY, USA, 2005. ACM.
- [YGW⁺12] Y. Yang, D. Günther, S. Wuhler, A. Brunton, H. P. Seidel, and T. Weinkauff. Correspondences of persistent feature points on near-isometric surfaces. In *Proceedings of the Fifth Workshop on Non-Rigid Shape Analysis and Deformable Image Alignment (NORDIA) in Proceedings of ECCV 2012 and its Workshops*, Florence, Italy, Oct. 2012.
- [YWS⁺06] L. Yin, X. Wei, Y. Sun, J. Wang, and M. J. Rosato. A 3D facial expression database for facial behavior research. In *Proceedings IEEE International Conference on Face and Gesture Recognition*, pages 211–216, 2006.
- [ZBVH09] A. Zaharescu, E. Boyer, K. Varanasi, and R. Horaud. Surface feature detection and description with applications to mesh matching. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 373–380, june 2009.
- [ZFN⁺95] H.-Q. Zhang, U. Fey, B. Noack, M. König, and H. Eckelmann. On the transition of the cylinder wake. *Physics of Fluids*, 7(4):779–795, 1995.
- [Zom01] A. Zomorodian. *Computing and Comprehending Topology: Persistence and Hierarchical Morse Complexes*. PhD thesis, Urbana, Illinois, October 2001.
- [ZSCO⁺08] H. Zhang, A. Sheffer, D. Cohen-Or, Q. Zhou, O. van Kaick, and A. Tagliasacchi. Deformation-driven shape correspondence. In *Proceedings of the Symposium on Geometry Processing*, SGP '08, pages 1431–1439, Aire-la-Ville, Switzerland, Switzerland, 2008. Eurographics Association.
- [ZWW⁺10] Y. Zeng, C. Wang, Y. Wang, X. Gu, D. Samaras, and N. Paragios. Dense non-rigid surface registration using high-order graph matching. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 382–389, june 2010.

List of Figures

2.1	Examples of cell complexes.	22
2.2	Examples of cell decompositions of smooth manifolds.	23
2.3	Examples of cubical complexes.	25
2.4	The lower level set $L_r(f)$ for different choices of r of an artificial height function f . The isovalue r increases from (a) to (f). The critical points of f are depicted in (f) as colored spheres: minimum (blue), saddle (yellow), maximum (red).	27
2.5	The ascending and descending manifolds of a terrain. The isolines are depicted as gray lines.	28
2.6	Illustration of the ascending and descending manifolds for a 3-dimensional input function. Arrows indicate the gradient flow. Blue and red spheres depict the minima and maxima while the saddles of index 1 and 2 are shown as green and yellow spheres, respectively. The 1-dimensional connections between two saddles are depicted as white lines. The ascending and descending manifolds decompose the domain into monotone regions.	29
2.7	Challenges of numerical algorithms.	30
2.8	The topology of the domain restricts the set of admissible Morse-Smale complexes. Discrete Morse theory provides a discretization of this set. The objective of this theory is to find a complex from this finite set of Morse-Smale complexes that represents the given scalar data well. . .	31
2.9	Simplicial collapse of a triangle to a vertex.	32
3.1	Illustration of a cell complex and its derived cell graph. a) shows the cells of a $2 \times 2 \times 2$ uniform grid in an exploding view. A single voxel is represented by eight 0-cells, twelve 1-cells, six 2-cells, and one 3-dimensional cell. These cells and their boundary relation define the cell complex. b) shows the derived cell graph. The nodes representing the 0-, 1-, 2-, and 3-cells are shown as blue, green, yellow and red spheres respectively. The adjacency of the nodes is given by the boundary relation of the cells. The edges are colored by the lower dimensional incident node. c) shows the cell complex and the cell graph to illustrate the neighborhood relation of the cells.	38
3.2	Illustration of a simplicial complex and its derived cell graph. (a) shows the cells of a 2-dimensional simplicial complex. (b) shows the derived cell graph. The nodes representing the 0-, 1-, and 2-cells are shown as blue, yellow and red spheres respectively. The adjacency of the nodes is given by the boundary relation of the cells.	39

3.3	Two graph representations of a 2-dimensional cubical complex. The nodes are colored by their index: blue ($k = 0$), yellow ($k = 1$), red ($k = 2$).	39
3.4	Morse matching: The edges of the Morse matching are shown as black solid lines. (a) shows a Morse matching of a 2D cell graph. The nodes of index 0, 1, and 2 are shown as blue, yellow, and red spheres. (c) shows a Morse matching of a 3D cell graph. The nodes of index 0, 1, 2, and 3 are shown as blue, green, yellow, and red spheres. (b) and (d) show a 0-streamline in the corresponding Morse matchings.	40
3.5	Illustration of the critical nodes and separation lines of a 2D Morse matching. The links of the Morse matching are depicted as solid black lines. The nodes of the cell graph in (a) are labeled by their index. The saddle and the minima in (b) are shown as yellow and blue spheres, respectively.	41
3.6	Properties of separation lines. The edges of the Morse matching are shown as solid black lines. Only the essential parts of the cell graph are shown. The nodes of the cell graph are labeled by their index. The saddles in (a) are shown as yellow spheres while the minimum is depicted as blue sphere. The repelling saddle in (b) is shown as green sphere while the two attracting saddles are shown as yellow sphere.	42
3.7	Comparison of running times for the analytic function g . a) The gray and yellow surfaces depict two different isolevels of the analytic function g . b) The circle and cross markers show the running times over the number of vertices n to construct the boundary matrix over different resolutions for Algorithm 6 and Algorithm 7, respectively. The semi-solid line depicts a least-square fitting of a linear function for the cross markers. The dotted line depicts a fitting of a quadratic function for the circle markers.	53
3.8	Illustration of Equation (3.3). The nodes of the graph G are labeled by their index. The critical points of the combinatorial gradient field V are shown as black spheres. Links belonging to V are depicted as solid black lines. The augmentation of V along the green path s results in a coarser gradient field $V \triangle s$ where the saddle (yellow) at one of its minimum (blue) are not critical anymore. A single minimum (blue dot) remains in $V \triangle s$	55
3.9	Cancellation of the saddle-minimum pair (s, m_2) . The minima, saddles, and maxima are depicted as blue, yellow, and red spheres, respectively. The 0- and 1-separation lines are shown as blue and red lines.	57
3.10	Synthetic noisy scalar field. Minima, saddles and maxima are depicted as blue, yellow and red spheres, while 0- and 1-separation lines are shown as blue and red lines, respectively.	59

3.11	Sketch of Algorithm 14. The descending manifolds of the 1-saddle (green sphere) are the 0-separation lines that connect the saddle to the minima (blue sphere). Firstly, the minimum-saddle pair with the smallest height difference is chosen. Secondly, the local intersection defined by the 1-saddle is computed. The ascending manifolds (blue surface) is integrated in a preprocessing step. This separation surface ends in 2-separation lines (red lines) that emanate from 2-saddles (yellow spheres). The 1-separation lines that connect the 1-saddle with the 2-saddles are shown as green lines. The right 2-saddle is connected twice with the 1-saddle. This saddle is therefore not a valid candidate for a simplification. From the remaining 2-saddles and the two minima the critical node is chosen that has the smallest weight with respect to the 1-saddle.	60
3.12	Performance of Algorithm 14. The red solid lines show the running time and the memory consumption for a uniform random field, the blue solid line for a distance field. The axes are shown in a loglog-scale. The full sequence \mathcal{V} was measured in all cases. The black dotted lines indicate a complexity of $O(n)$ and $O(n^{5/3})$	65
3.13	Depiction of a combinatorial gradient field in a subgraph of G connecting 2-saddles (yellow) and 1-saddles (green). The critical nodes are labeled with their assigned scalar value. The links of the matching are depicted as black solid lines.	66
3.14	Graph of the weights of the augmenting paths. (a) shows the weights of augmentations for the artificial function g (3.1) over the number of augmentations. The weights behave monotonically increasing. The monotony is broken in (b) where a small amount of noise is added to g . The number of monotony breaks further increases if the level of noise is increased (c).	66
3.15	Illustration of the connectivity change due to a saddle-saddle simplification. Shown is a subgraph of G connecting 2-saddles (yellow) and 1-saddles (green). The links of the matching are depicted as black solid lines. The blue line in (a) depicts an augmenting path. After the augmentation, the connectivity of the saddles (red lines) completely changed (b).	67
3.16	Illustration of different levels of details of \mathcal{V} of the analytic function g (3.1). Minima, 1-saddles, 2-saddles and maxima are depicted as blue, green, yellow and red spheres, respectively. The p -separation lines are shown as blue ($p = 0$), green ($p = 1$) and red ($p = 2$) lines. The iso-surface (grey) in (c) illustrates the most dominant minima and maxima regions. The hierarchy consists of 207 levels.	68
4.1	Persistent homology of a 1D function $f(x)$. The persistence pairs consist of (x_1, x_4) and (x_3, x_2) . The persistence of x_1 and x_4 is therefore given by $f(x_4) - f(x_1)$, while the persistence of x_2 and x_3 is given by $f(x_2) - f(x_3)$	72

4.2	Distance field of a Chaperon protein. An isosurface of a distance field, computed from a protein, as gray transparent surface is shown in image a). The maxima and the 2-saddles are shown as red and yellow spheres, respectively. Each sphere is scaled by its persistence. Image b) shows the persistence diagram of the data set containing all dimensions. The axes denote the data range of the distance field.	76
4.3	Simplification increases the length (top row) and area (bottom row) of affected separation lines and surfaces. Shown are the topological structures before (left) and after (right) a simplification.	80
4.4	Persistence-based simplification. Shown are the extremal structures of the initial field V_0 (left) and the final field V_m (right), which still contains critical points. Gray isosurfaces illustrate the underlying synthetic function.	81
4.5	Matching results for different models, where correspondences are shown in the same color and connected by a line.	82
4.6	Matching results for different models corrupted by synthetic noise. From left to right: outliers, holes, topological noise, partial information, and Gaussian noise.	85
4.7	(a) Visualization of dense correspondence results. The same color is mapped from one surface to another using the dense correspondences. (b) Tracking of the <i>Eagle</i> sequence. Tracked features are shown as colored curves.	86
4.8	Matching between different object classes.	88
4.9	(a,b): Influence of parameter values on matching two clean <i>Cat</i> models. The x -axes show the thresholds and the y -axes show the number of matches. (c,d): Correspondence error \mathcal{C} for the <i>Cat</i> model with different types and levels of degradation. Each bar in (c) is the mean over all model pairs, while each bar in (d) is for one pair.	89
4.10	Comparison for <i>Cat</i> without noise and with second level of holes. . .	90
4.11	Correspondences of scans with different facial expressions. The top row shows the textured raw scans and the bottom row shows our results. . .	91
5.1	Illustration of the analytic function f (5.1): The minima, saddles and maxima are depicted as blue, yellow, and red spheres. The 0- and 1-separation lines are shown as blue and red lines, respectively. The integral lines of the gradient ∇f are illustrated by green and blue arrow lines in (a). The local attracting and repelling behavior of a separatrix can change as illustrated by the central separatrix in (b). The bending of the gray isolines in (b) depicts a change from a local convex to a local concave behavior of f . However, the gradient flow is still separated by the separatrix (central red line).	94
5.2	Extremal lines of the scalar field (3.1) following two different definitions. . .	96
5.3	Separatrix persistence assigns an importance weight to each point along a separatrix. It respects that the importance of a feature may smoothly change along the line.	97
5.4	The cancellation of (s, m_2) assigns a final persistence to the removed separatrices (gray, dashed). Neighboring separatrices gain persistence (red, thick).	98
5.5	Illustration of the local feature strength of separation surfaces. The evolution of isocontours is depicted as yellow surfaces.	100

5.6	Illustration of the local feature strength of separation lines. The evolution of isolines is depicted as black lines.	100
5.7	Extremal lines scaled by local feature strength (left) and separatrix persistence (right). Small fluctuations in the data cause an improper representation of the dominant extremal structures using the local feature strength. Separatrix persistence, in contrast, reveals the global structure.	101
5.8	Human torso. (left) $\kappa_{max}/\kappa_{min}$ of the surface shown in front/back. (middle) Reduction of the result size by removing lines with very low persistence (green); exemplified for the minimal lines of κ_{min} . Kept lines are shown in black and scaled according to separatrix persistence. (right) Perceptually salient convex edges are shown in red, concave edges in blue.	104
5.9	Results from left to right: dinosaur, screwdriver, knot model, plot of κ_{min} and concave edges of the brain model, blade.	105
5.10	Comparison of methods regarding their robustness to noise. All methods had to deal with the raw input, i.e., any smoothing of the surface or the curvature was forbidden. Curvatures have been computed in the default way for the respective method, i.e., [HPW05] uses its discrete shape operator, while our method and the SC package use normal variation. We used a clipping plane in the last two rows to expose the feature line. As it can be seen, our method is less noise-sensitive thanks to its global nature.	106
5.11	Comparison of methods regarding parameter dependence and the connectedness of the results using the feline model. Compared to ridge/valley-based methods, our technique yields qualitative similar results without derivatives and computational parameters.	107
5.12	Comparison of valley networks: a) and b) show the valley networks manually mapped by Carr [Car95] and Hynek et al. [HP03] as orange and yellow lines. c) shows our automatic extraction result as blue lines (scaled by separatrix persistence).	108
5.13	Different levels of detail of extremal lines: For a given elevation map, a binary mask was computed to mask the craters. Four levels of the hierarchy are depicted – the initial topological structures (where all extremal lines are present) and three less detailed levels are shown. The width of the extremal lines is scaled by separatrix persistence.	109
5.14	Extraction of maximal lines in a CT-scan of an aneurism using separatrix persistence.	110
5.15	Height ridges of the Bonsai data set.	111
5.16	Maximal lines of the Bonsai data set.	111
5.17	3D unsteady flow behind a cylinder. Shown are extremal features of the Q -criterion for $t = \pi$. The gray isosurface depicts the zero-level of Q while the level $Q = 2.7$ is illustrated as yellow isosurface.	112
5.18	FTLE of the ABC Flow. Filtering the separation surfaces of the Morse-Smale complex using separatrix persistence reveals the most dominant surfaces of maximal FTLE value.	113
5.19	Discrete (top) and smooth (bottom) representation of a subregion of the cell membrane.	114

5.20	Extraction of a cell membrane in a cryo-electron tomogram using separatrix persistence (left column) and ridge/valley definition (right column). The top row shows the original data. The bottom row shows the results after Gaussian smoothing.	114
6.1	Separation lines and surfaces of the electrostatic field of a benzene molecule for V_{m-85} with 181 critical points. (a) shows the minimal structures: the 48 minima (blue spheres), 78 1-saddles (green spheres), 0-separation lines (blue lines) and the ascending manifold of the 1-saddles (blue surface). (b) shows the maximal structures: the 12 maxima (red spheres), 43 2-saddles (yellow spheres), 2-separation lines (red lines) and the descending manifolds of the 2-saddles (red surface). Due to symmetry, only one half of the separation surfaces is shown.	115
6.2	Comparison of combinatorial and continuous extremal structures for the electrostatic field around a benzene molecule. (a) shows smooth extremal structures extracted as in [TWHS03a]. The minima and the maxima are depicted as blue and red spheres while the 1- and 2-saddles are shown as blue and red disks respectively. The 1-separation lines are shown as blue-red stripes. Gray illuminated lines represent streamlines emanating from the saddles. (b) shows combinatorial extremal structures. The minima, 1- and 2-saddles, and the maxima are represented by blue, green, yellow and red spheres respectively. The 1-separation lines are shown as green lines. Gray illuminated lines depict the 2-separation lines emanating from the 2-saddles. (c) and (d) show a close-up of the center 2-saddle with its surrounding 1-saddles and the corresponding 1-separation lines for the smooth and combinatorial case. Gray surfaces depict the carbon and the hydrogen atoms and their bonds.	116
6.3	Illustration of the steepest descent of a locally constant gradient (black diagonal lines). The gradient is projected on a uniform grid (gray). The steepest descent at the vertices of the grid (black arrows) always points north.	118
6.4	The analytic example f (6.6). The first row shows the sampled function f color-coded for different choices of α . Red denotes a high function value, while blue denotes a low value. Black lines depict integral lines of the gradient ∇f . The second and third row show the separation lines based on Algorithm 5 using the steepest descent and our probabilistic link selection strategy, respectively. Minima, saddles and maxima are shown as blue, yellow and red spheres, whereas the 0- and 1-separatrices are shown as blue and red lines, respectively.	120
6.5	The analytic example f (6.6). (a) and (b) show the Hausdorff distance of the approximated center circle to the reference circle for different choices of α and increasing resolution. (a) shows the evolution of the error using the steepest descent approach. (b) shows two statistical characteristics of the error of the probabilistic approach.	121
6.6	Distribution of Hausdorff distance error. The blue, black and red curves show the estimated probability density function of the Hausdorff distance between the center circle and the reference circle (as shown in Figure 6.4) for different resolutions.	122

- 6.7 Comparison of the steepest descent and the probabilistic approach using generic functions. The four rows show four different functions randomly generated using expression (6.7). Red denotes a high function value, while blue denotes a low value. Black lines depict integral lines of their gradient. The left two columns show the extracted Morse-Smale complexes for each function sampled on a 512^2 grid using the steepest descent and our probabilistic link selection strategy, respectively. The right two columns show the result on a 4096^2 grid. Minima, saddles and maxima are shown as blue, yellow and red spheres, whereas the 0- and 1-separatrices are shown as blue and red lines, respectively. Since separatrices are integral lines of the gradient, the blue and red lines should follow the black lines. White arrows indicate regions where the difference of the two approaches is visually apparent. 123
- 6.8 Visualization of the electrostatic potential around the benzene molecule. 126

List of Tables

2.1	Examples of homologically different complexes.	24
3.1	Algorithmic overview of the computational framework.	36
3.2	Overview of the notations used in the algorithms.	37
3.3	Running times of Algorithm 5 for six data sets of varying dimensions. The first row shows the running time to compute the combinatorial gradient field using single threaded computation while the second row shows the times using the 4 physical and 8 logical cores. The resulting speed up factor is shown in the third row.	46
3.4	Running time of Algorithm 11.	59
3.5	Running times of Algorithm 14. The first and second rows show the running times for the computation as well as the number of levels of a 5% and a complete simplification, respectively.	69
4.1	Running times and memory consumption for 3D images of different size and topological complexity. The second column shows the topo- logical properties of the data sets. The third column shows the total memory consumption of our method compared to Wagner et al. [WCV12] using 1 and 24 logical cores. The total running time for the construc- tion of the initial gradient field, the boundary matrix and the matrix reduction using 1 and 24 logical cores are compared to the times of Wagner et al. [WCV12] in the fourth column.	79
4.2	Parameter settings for all models used in the tests.	87

Index

algorithms

- AlternatingEdges, 43
- AlternatingRestrictedBFS, 43
- BoundaryMatrix, 48
- CombinatorialGradient, 45
- CountPaths, 49
- GetAllManifolds, 49
- GetIntersection, 44
- GetManifold, 43
- GetManifoldNodes, 50
- GetMinWeight2D, 58
- GetMinWeight3D, 62
- GetPath2D, 58
- GetPath3D, 62
- InitHeap, 61
- IsUnique, 63
- notations, 37
- overview, 36
- Robins et al., 48
- Sequence2D, 56
- Sequence3D, 60

Betti number, 24, 26

cell, 22

- coface, 23
- critical, 32
- face, 23

cell complex, 22

- cubical, 25, 39
- lower subcomplex, 32
- simplicial, 25, 39

cell graph, 38

- links, 39
 - index, 39
- nodes, 38
 - index, 38

chain, 23

- boundaries, 24
- coefficients, 23
- cycles, 24

combinatorial

- separation line, 41
 - properties, 42, 44
- separation surface, 41
- streamline, 41

critical

- node, 41
- point, 26
 - index, 26
 - non-degenerated, 26

CW-decomposition, 22

extremal structures

- crest lines, 105
- extremalities, 105
- Gaussian curvature extrema, 84
- height ridges, 95
- local vs. global, 94
- ridges/valleys, 105
- separatrices, 95

function

- discrete Morse, 31
- filtering, 73
- lower level set, 26
- lower-star filtration, 45
- Morse, 26
- Morse-Smale, 28

gradient, 26

- combinatorial, 41, 45
 - hierarchy, 54, 56, 59
 - properties, 45, 65
- negative flow, 27, 42

Hasse diagram, 39

Hessian, 26

homology group, 24

- persistent, 73

isometry, 84

- manifold
 - ascending, 27
 - descending, 27
- map
 - attaching, 22
 - boundary, 23, 39, 47
- matching, 40
 - augmentation, 54
 - Morse, 41
- Morse-Smale complex, 29
 - critical points, 29
 - discrete, 47
 - convergence, 118
 - separation line, 29
- persistence, 55, 73, 84
 - algebraic, 74
 - and simplification, 80
 - of a separatrix, 96, 124
 - 2D, 99
 - 3D, 101, 102
 - output-sensitive, 75
 - pairs, 73
 - stability, 74
- simplicial collapse, 32
- symmetric difference, 23, 54
- theorem
 - cancellation, 55
 - discrete Morse theory, 32
 - isolated critical point, 26
 - Morse inequalities, 26
 - Morse theory, 29
 - optimal Morse matching, 65
- transversal, 28