Contributions to Topological Data Analysis for Scientific Visualization

Julien Tierny



Sorbonne Universités, UPMC Univ Paris 06

Laboratoire d'Informatique de Paris 6

Habilitation à Diriger des Recherches

Spécialité Informatique

par

Julien Tierny

Contributions to Topological Data Analysis for Scientific Visualization

Habilitation soutenue le 29 Avril 2016 devant le jury composé de :

Mme	Isabelle Bloch	Professeur, Télécom ParisTech	(Examinateur)
M.	Jean-Daniel Fekete	Directeur de Recherche, INRIA	(Examinateur)
M.	Pascal Frey	Professeur, UPMC Sorbonne Universités	(Examinateur)
M.	Hans Hagen	Professeur, Technische Universität Kaiserslautern	(Rapporteur)
M.	Chris Johnson	Professeur, University of Utah	(Rapporteur)
M.	Bruno Lévy	Directeur de Recherche, INRIA	(Rapporteur)
M.	Philippe Ricoux	Direction Scientifique, Total	(Examinateur)
M.	Will Schroeder	Directeur, Kitware Inc.	(Examinateur)

Sorbonne Universités, UPMC Univ Paris 06
Laboratoire d'Informatique de Paris 6
UMR UPMC/CNRS 7606 – Tour 26
4 Place Jussieu – 75252 Paris Cedex 05 – France

This manuscript is dedicated
to all those who supported me along the years.

" Any problem which is non-linear in character, which involves more than one coordinate system or more than one variable, or where structure is initially defined in the large, is likely to require considerations of topology and group theory for its solution. In the solution of such problems classical analysis will frequently appear as an instrument in the small, integrated over the whole problem with the aid of topology or group theory. "

Marston Morse, 1934 (Mor34)

Foreword

This manuscript reviews my research work since my Ph.D. thesis defense (2008), as a post-doctoral researcher at the University of Utah (2008-2010) and a permament CNRS researcher at Telecom ParisTech (2010-2014) and at Sorbonne Universités UPMC (2014-present).

This work includes results obtained in collaboration with several research groups (University of Utah, Lawrence Livermore National Laboratory, Lawrence Berkeley National Laboratory, Universidade de Sao Paulo, New York University, Sorbonne Universites, Clemson University, University of Leeds) as well as students whom I informally or formally advised.

This research has been partially funded by several grants, including a Fulbright fellowship (US Department of State), a Lavoisier fellowship (French Ministry for Foreign Affairs), a Digiteo grant (national funding, "Uncertain Topo-Vis" project 2012-063D, Principal Investigator), an ANR grant (national funding, "CrABEx" project ANR-13-CORD-0013, local investigator), a CIFRE partnership with Renault and a BPI grant (national funding, "AVIDO" project, local investigator).

During this period, I taught regularly at the University of Utah (2008-2010), Telecom ParisTech (2011-present), Sorbonne Universités (2011-present) and since 2013 at ENSTA ParisTech and University of Versailles, where I am the head instructor for the scientific visualization course.

This manuscript describes most of the results published over this period (chapter 4: (TGSP09, TP12), chapter 5: (TGSP09, GGL*14, TDN*12), chapter 6: (STK*09, BWT*09, BWT*11, GABCG*14) chapter 7: (TP12, TDN*12, TDN*11, STP12, PST*13, PST*15, ENS*12) chapter 8: (CGT*15, KTCG15, GST14)). I refer the interested reader to the following publications (TBTB12, TTB12, TTB13, MTT*13, GTJ*13) for additional results not described in this document.

The reading of this manuscript only requires basic background in Computer Science and algorithmic; most of the mathematical notions are introduced in a dedicated chapter (chapter 3).

Chapter 2 provides a short summary of the entire manuscript.

Acknowledgments

First, I am sincerely grateful to Isabelle Bloch, Jean-Daniel Fekete, Pascal Frey, Hans Hagen, Chris Johnson, Bruno Lévy, Philippe Ricoux, and Will Schroeder, who did me the honor of accepting to be part of my Habilitation committee. I am especially thankful to the reviewers of my manuscript for accepting this significant task. Second, I would like to thank all my collaborators over the last seven years. The results presented in this manuscript would not have been possible without them. Since my Ph.D. defense, I had the opportunity to work with more than forty coauthors, and I like to think that I learned much from each one of them. More specifically, I would like to thank some of my main collaborators (in alphabetical order), hoping the persons I forgot to mention will forgive me: Timo Bremer, Hamish Carr, Joel Daniels, Julie Delon, Tiago Etiene, Attila Gyulassy, Pavol Klacansky, Josh Levine, Gustavo Nonato, Valerio Pascucci, Joseph Salmon, Giorgio Scorzelli, Claudio Silva, Brian Summa[1], Jean-Marc Thiery. Along the years, some of these persons became recurrent collaborators with whom I particularly enjoyed working and interacting. Some of them even became close friends (even best men!) and I am sincerely grateful for that. Special thanks go to Valerio Pascucci, who gave me a chance back in 2008 when he hired me as a post-doc, although we had never met before. I have no doubt that my career path would have been very different if we had not worked together. Working with Valerio and his group has been a real pleasure and a source of professional and personal development. I am both glad and proud to be able to say that our collaboration lasted well beyond my stay at the University of Utah and that it still continues today. Along the last seven years, Valerio has been a careful and inspiring mentor and I am sincerely grateful for that. Next, I would like to thank all of the colleagues I had the chance to interact with at the University of Utah, Telecom ParisTech and Sorbonne Universites UPMC, in particular my current students (Ana, Charles and Guillaume) who are a daily source of motivation. Finally, I would like to thank my friends, my family, and my wife for their constant love and support.

[1]Since you literally and so kindly asked for it, there you go buddy :)

Contents

1 Introduction

In early 2013, a group of researchers led by French scientists published in Nature a paper entitled *"A vast, thin plane of corotating dwarf galaxies orbiting the Andromeda galaxy"* (ILC*13). This paper reported new intriguing observations that showed that a majority of the dwarf galaxies which orbit the larger Andromeda galaxy was actually rotating in a very thin, common plane structure. These observations then contradicted the state-of-art models which assumed that dwarf galaxies' locations followed an isotropic random distribution. This discovery raised many fundamental open questions that can potentially reshape the entire understanding of the universe formation process, as it implies that a still-to-be-found phenomenon seems to control the geometry of cosmos gas flow.

Beyond its academic outreach, this work drew a lot of attention from the French media, as one of the co-authors of the paper was a French teenager (and probably one of the youngest co-authors of a Nature publication). This student was doing a summer internship in a French astrophysics laboratory where he was assigned the design of a simple software prototype for the visualization of dwarf galaxy measurements. This is only when they started to visualize these measurements in 3D that these researchers made the astonishing observation of a coplanar orbit distribution, an hypothesis that was later confirmed through numerical estimations. In this study, while the numerical verification of the co-planarity hypothesis can be considered as a trivial task, formulating the original idea of this hypothesis cannot. Here, simple visualization tools precisely enabled this initial discovery as they helped these researchers formulate such original insights about their data.

This anecdote effectively illustrates one of the key motivations of Scientific Visualization, which is a sub-field of Computer Science that aims at developing efficient algorithms for the graphical and interactive exploration of scientific data, for the purpose of hypothesis formulation, analysis and interpretation.

While galaxy orbits are made of moderately simple geometries, recent acquisition devices or high-performance computing simulations nowadays generate large-scale data-sets of extremely precise resolution, which can encompass features with highly complex geometry, challenging their visualization and analysis. Therefore, research in Scientific Visualization aims at addressing several general challenges which impact distinct stages of the scientific methodology:

1. **Abstraction:** The definition of efficient analysis algorithms able to abstract high-level features (that humans can visualize, measure and understand) from raw data;

2. **Interaction:** The definition of efficient algorithms for the interactive manipulation, simplification and exploration of these high-level features;

3. **Analysis:** The definition of efficient algorithms for the geometrical measurement of these features, to serve as base tools for interpretation tasks in specific application problems.

Regarding scalar valued data, Topological Data Analysis form a family of techniques that gained an increasing popularity in the Scientific Visualization community over the last two decades, since it precisely enables the robust capture and multi-scale representation of geometrical objects that often directly translate into features of interest application wise.

In this manuscript, I review the main results of my research over the last seven years in this area, where I contributed to Topological Data Analysis in each of the topics described above (abstraction, interaction and analysis). I also discuss research perspectives for Topological Data Analysis as well as preliminary results regarding the analysis of multivariate and uncertain data.

The rest of the manuscript is organized as follows:

- Chapter 2 presents a short summary of the entire manuscript;

- Chapter 3 describes the theoretical background of Topological Data Analysis and briefly reviews the state-of-the-art;

- Chapter 4 describes my contributions to the problem of defining efficient algorithms for the computation and simplification of topological abstractions of scalar data;

- Chapter 5 describes my contributions to the problem of defining efficient algorithms for interacting with topological data abstractions;

- Chapter 6 describes my contributions to the problem of defining efficient analysis algorithms of topological abstractions in specific application driven problems;
- Chapter 7 describes my contributions to related problem which I addressed with solutions derived or inspired from Topological Data Analysis;
- Chapter 8 describes my view on the perspectives and upcoming challenges for Topological Data Analysis and includes preliminary results regarding the analysis of multivariate and uncertain data.
- Chapter 9 finally concludes this manuscript.

Summary

2

Contents

This chapter summarizes the entire manuscript and presents a summary of my activities since my Ph.D. thesis defense in October 2008.

First, I summarize as a starting point the main results of my Ph.D. thesis. Then my research activity as a post-doctoral researcher and a permanent researcher is reviewed. This report summarizes my main results as well as the perspectives of my research.

Second, I briefly describe my teaching activities and responsibilities.

2.1 Research activity

My research deals with the algorithmic aspects of the analysis of discrete geometrical data. Specifically, my core expertise lies in the topological analysis of scalar data. The addressed problems are mostly application driven and a strong emphasis is given on the practical applicability of the designed algorithms with specific applications in mind, in particular to scientific visualization.

2.1.1 Ph.D. thesis summary

My Ph.D. thesis, entitled "Reeb graph based 3D shape modeling and applications" (Tie08), dealt with the geometrical analysis of triangulated surfaces for shape modeling and comparison. It was supervised by Pr. Jean-Philippe Vandeborre and Pr. Mohamed Daoudi of the 3D SAM team of Lille1 University (France), a research group dedicated to pattern recognition and shape analysis.

My main Ph.D. research results include new automatic algorithms for the computation of skeletal shape representations based on the Reeb graph. I applied these algorithms in several problems related to Computer Graphics, including skeletonization for shape deformation (TVD06b, TVD06a, TVD08a), shape segmentation (TVD07b), animation reverse engineering (TVD08b) and shape comparison (TVD07a, TVD09). In that latter topic, I developed a new approach for the partial comparison of triangulated surfaces which outperformed competing approaches on the international benchmark SHREC 2007.

During this thesis, I gained a strong knowledge in topological data analysis, as I was a user of these tools for my own algorithms.

2.1.2 Environment

After my Ph.D. defense, I wanted to develop my expertise in topological data analysis and apply these techniques to a different class of problems. I joined the Scientific Computing and Imaging Institute at the University of Utah in late 2008 as a Fulbright scholar. There, I considerably strengthened my expertise in topological techniques, addressed harder problems dealing with more complex data (in geometry, dimension and size) and started to explore the field of Scientific Visualization.

I joined the CNRS in late 2010 as a permanent researcher in the image

processing group of Telecom ParisTech, where I continued my research at the interface between Computer Graphics and Scientific Visualization.

Since September 2014, I am part of the Scientific Computing department of the Computer Science laboratory of Sorbonne Universités UPMC, which offers a unique environment to develop research activities in Scientific Visualization thanks to its strong inter-disciplinary collaboration culture.

2.1.3 Main results

Since 2008, my research focused on the definition of efficient algorithms for the computation of topological abstractions of scalar data as well as efficient algorithms for their exploitation in practical problems. The related results can be classified in four categories, described in the following:

1. *Abstraction* (chapter 4);
2. *Interaction* (chapter 5);
3. *Analysis* (chapter 6);
4. *Related Problems* (chapter 7).

Abstraction

In this topic, I focused on the definition of algorithms with possibly non-optimal time complexity but with efficient practical behavior for the construction of topological abstractions or the topological simplification of scalar data. This work is described with further details in chapter 4.

The time complexity of most algorithms for the construction or processing of topological abstractions is dictated by the number of critical points in the input scalar field. Often in practice, it is possible however to easily discriminate critical points that are not relevant application-wise. Therefore, there exists an applicative interest for an efficient pre-processing of an input scalar field, that would minimally perturb it to remove a given set of critical points. We introduced such a technique in 2012 for piecewise linear scalar fields defined on surfaces (TP12). This approach is based on a new iterative algorithm for the constrained reconstruction of sub- and sur-level sets. Experiments showed that the number of iterations required for our algorithm to converge is rarely greater than 2 and never greater than 5, yielding $O(nlog(n))$ practical time performances. Thanks to its simplicity, ease of implementation, speed, robustness and generality,

we consider this algorithm as the reference for the problem of topological simplification of scalar data on surfaces.

The Reeb graph (Ree46) has been a popular topological abstraction for the efficient indexing of the connected components of level sets of scalar data, for feature extraction, segmentation and exploration purposes. While an algorithm with optimal time complexity existed for its construction on surfaces (CMEH*03), for volumes, algorithms with practical quadratic behavior were only available. We introduced an algorithm called *loop surgery* (TGSP09) that, given an input scalar field, modifies the volume by a sequence of cuts guaranteeing the resulting Reeb graph to be loop free. Thus, the loop surgery reduces the Reeb graph computation to the simpler problem of computing a contour tree, for which well known algorithms exist that are theoretically efficient and fast in practice (CSA00). While the loop surgery procedure has a quadratic worst case complexity, experiments showed in practice that our overall approach achieved virtually linear scalability and outperformed competing approaches by up to 3 orders of magnitude. We considered this algorithm as the reference for the problem of Reeb graph computation on volumes until an optimal time complexity algorithm was introduced three years later (Par12).

Interaction

In this topic, I focused on the definition of algorithms capable of interacting with topological abstractions. This includes simplification mechanisms as well as atomic editing operations for user driven data segmentation purposes. This work is further described in chapter 5.

When dealing with noisy data sets, level set extraction can yield many connected components that prevent a clear visualization of the main components of the isosurface due to occlusion. We illustrated how the Reeb graph could be used as a query data-structure to extract isosurfaces in optimal time and how it could be simplified according to a user provided metric to progressively remove connected components considered as noise. While such a process was known in the community for a long time, our Reeb graph computation scheme for volumes (TGSP09) enabled such a capability for the first time at interactive rates.

Based on this approach, we introduced next an isosurface based visualization widget for a large-scale simulation monitoring system. Large scale numerical simulations running on super-computers are often analyzed online with advanced monitoring systems (KVP*08), providing the end users

with real time quantitative indicators describing the behavior of the simulation, allowing them to identify instantly possible run-time issues (mistakes in parameter settings for instance). However, the interpretation of these indicators require strong user expertise and sometimes problematic configurations are difficult to read from those. Therefore, there exists an application need for on-the-fly visualizations of numerical simulations for monitoring purposes. However, large scale simulation time steps are usually too large to be interactively transfered to a remote workstation for visualization. Moreover, due to their size, not all of the time-steps of such a simulation can be stored on the super-computer where they have been generated. This prevents remote rendering and interaction and requires in-situ visualization generation. We addressed this problem by designing a prototype (STK*09) capable of generating in-situ isosurface renderings. The visualization of an isosurface is dictated by a number of parameters such as view point and isovalue and possibly topological simplification threshold. Therefore we implemented an algorithm that finely samples in batch mode this parameter space and generates for each parameter combination an in-situ offline 2D rendering. Even by finely sampling this parameter space, the data size of the output collection of 2D renderings for a given time-step is still guaranteed to be orders of magnitude smaller than that of the time-step itself, allowing a remote emulation of the interactive control of these parameters. This technique was enabled by our fast isosurface extraction and simplification algorithms described previously. This prototype enabled simulation users to obtain for the first time qualitative visual insights from their running simulations. A similar strategy has been developed independently by Kitware Inc. five years later in its visualization system ParaView Cinema (Kit14).

The Morse-Smale complex has been a popular topological abstraction for the efficient indexing of gradient integral lines in scalar data, for feature extraction, exploration and segmentation. In many application scenarios, the domain decomposition it provides directly corresponds to meaningful segmentations application-wise with excellent classification scores. However, such an approach still results in general in the identification of false negatives as compared to a manual labeling by a domain expert. Therefore, we derived a combinatorial approach (GGL*14) to perturb the input scalar field's discrete gradient according to some prior segmentation knowledge (either obtained interactively or automatically). This algorithm guarantees that the input segmentation constraints are captured as separatrices in the Morse-Smale complex. Such a framework enables the

incorporation of automatic or interactive prior knowledge in Morse-Smale complex based segmentation, while still benefiting from its automatic and multi-scale nature. In practice, this enables to automatically extend sparse user knowledge into correct segmentations.

In Computer Graphics, surface quadrangulation representations are often preferred over triangulations for tasks such as texture mapping or animation. Quadrangulations can be obtained by partitioning the surface in a set of quadrangular charts (which can be further refined as desired). However, end-users need to control the overall layout and topology of this partitioning (chart number and boundary alignment, position, valence and number of extraordinary vertices) as it can affect the output animation. We introduced a framework for the user-driven quadrangular segmentation and parameterization of surfaces called Reeb atlases (TDN*12), which provides an explicit and robust control on the configuration of extraordinary vertices. It is based on a number of atomic operations defined on the Reeb graph of an underlaying harmonic scalar field. These operations include the insertion, deletion, subdivision and merging of charts as well fractional critical point editing. The latter capability was based on an algorithm for the definition and control of highly degenerate configurations usually avoided in topological data analysis. Experiments demonstrated the interactivity and flexibility of the approach, as well as its ability to generate quadrangulations with high quality statistics, while robustly fitting the user defined constraints.

Analysis

In this topic, starting from precise application problems, I explored how algorithms from topological data analysis could be adapted to conduct interactive data exploration and quantitative analysis. This work is described with further details in chapter 6.

The interpretation of numerical simulations often requires the quantitative analysis of features of interest. In the context of turbulent combustion simulations, we developed a framework based on the split tree (a variant of the Reeb graph) to extract, enumerate and track flames through time (BWT*09, BWT*11). This approach is based on a multi-scale segmentation of the data into connected components of sur-level sets of fuel consumption rates. Such an approach enables to segment the main flames of the simulation and to track them through time. We built an interface on top of this algorithm capable of visualizing these flames as well as their

temporal evolution. Moreover, our approach enabled the computation of quantitative metrics such as flame volume and life-span. Such quantitative indicators helped the end users interpret their simulation to identify and characterize distinct combustion regimes in the vicinity of the burner.

In chemistry, the interactions between atoms have a major influence on the chemical properties of molecular systems. While covalent interactions impose the structural integrity of molecules, noncovalent interactions govern more subtle phenomena such as protein folding or molecular bonding. The understanding of these types of interactions is necessary for the interpretation of many biological processes and chemical design tasks. While the extraction and characterization of covalent interactions can be performed by a topological analysis of the electron density, noncovalent interactions are characterized by low electron densities and only slight variations of them – challenging their extraction and characterization. To address this problem, we presented the first combinatorial algorithm for the automated extraction and characterization of covalent and noncovalent interactions in molecular systems (GABCG*14). The proposed algorithm is based on a joint topological analysis of the signed electron density and the reduced gradient. Combining the connectivity information of the critical points of these two scalar fields enables not only to identify automatically these interactions but also the atoms and bonds involved in each localized interaction. Experiments on a variety of molecular systems, from simple dimers to proteins found in DNA, demonstrated the ability of our technique to robustly extract these interactions and to reveal their structural relations to the atoms forming the molecules. For simple systems, our analysis corroborated the observations made by the chemists while it provided new visual insights on chemical interactions for larger molecular systems.

Related Problems

In this topic, I explored related problems that have been addressed with a solution derived or inspired from topological data analysis. This work is described with further details in chapter 7.

Many geometry processing problems involve numerically sensitive tasks such as partial differential equation resolution, gradient field integration, or scale-space computation. In many cases, the topology of the numerical solution is a major consideration. In meshing for instance, extraordinary vertices often correspond to singularities and these important

constraints must be respected. However, numerical noise often occurs and it can alter the topology of the solution. This is the case for instance for the Laplace equation subject to Dirichlet boundary conditions. Beyond its ubiquity in geometry processing, this equation plays an important role in electromagnetism, astronomy and fluid dynamics. Given a finite set of extrema constraints along with corresponding target values, the solution to this equation is a piecewise linear scalar field with prescribed values at the constraints and a zero-valued Laplace operator everywhere else. An important property of this equation is that the Dirichlet constraints should be the only extrema of the solution. However, since the Laplacian is a second-order operator, it is difficult to discretize for piecewise linear functions. Hence several discretization strategies have been proposed (WMKG07). The popular discretization based on cotangent weights (PP93) is usually preferred since it produces smooth level sets. However, in practice, the numerical sensitivity of the cotangent operator induces numerical noise which may generate additional critical points, which prevents the solution from conforming to its formal description. We presented a simple technique based on our scalar field simplification algorithm (TP12) that automatically perturbs this solution to only admit extrema on the Dirichlet constraints. Therefore, our approach can be used to generate a solution with both the geometrical accuracy of the cotangent weight Laplacian and the topological robustness of the combinatorial Laplacian, yielding a solution with topological guarantees that is exploitable for *certified* geometry processing.

This latter result served as a key ingredient for the robust editing of the Reeb graph in our work on surface quadrangulation with Reeb atlases (TDN*12, TDN*11). In addition to its interactive quadrangular segmentation and parametrization capabilities, we further enriched this approach by allowing the user to refine the connectivity of each quadrangular chart with the notion of connectivity texture, which could be either manually designed (TDN*12) or automatically retrieved from a collection of predesigned quadrangulation and automatically fitted on the input surface (TDN*11).

A fundamental step in stitching several pictures to form a larger panorama is the computation of boundary seams that minimize the visual artifacts in the transition between images. Current seam computation algorithms use optimization methods that may be slow, sequential, memory intensive, and prone to finding suboptimal solutions related to local minima of the chosen energy function. Moreover, even when these techniques

perform well, their solution may not be perceptually ideal (or even good). Such an inflexible approach does not allow the possibility of user-based improvement.

Surprisingly, in this problem, the analysis of the topology of the layout of registered images helps in selecting relevant subsets of pairwise image boundaries to be included in the overall seam network. Based on this analysis, we introduced a technique named Panorama Weaving (STP12) for seam creation and editing in image panoramas. It provides a procedure to create automatically boundaries for panoramas that is fast, has low memory requirements and is easy to parallelize. This technique often produces seams with lower energy than the competing global technique, while running orders of magnitude faster. Second, it provides the first interactive technique for the exploration of the seam solution space. This powerful editing capability allows the user to automatically extract energy minimizing seams given a sparse set of constraints. A variety of empirical results showed the importance of fast seam computation for interactivity and the usefulness of seam space exploration for the correction of visual artifacts. We consider this algorithm as the reference for the problem of seam creation and editing in panoramas. This approach served as the basis for our results on the automatic seam computation for large scale panoramas (PST*13, PST*15).

Scientific Visualization has become a standard component in scientific software. However, unlike traditional components of the scientific pipeline (such as mathematical modeling or numerical simulation), only few research efforts have been devoted to the verification of the accuracy, reliability and robustness of its algorithms. Inspired by Morse Theory and Topological Data Analysis, we introduced a framework for the topological verification of isosurface extraction implementations based on the trilinear interpolant (ENS*12). In particular, we developed simple algorithms based on stratified Morse theory and digital topology to evaluate the topological invariants of an isosurface (Euler characteristic and Betti numbers) to compare them with these of the actual surface meshes returned by the implementations under evaluation. This methodology revealed unexpected behaviors and even coding mistakes in publicly available popular isosurface codes. In addition to prior work on geometrical verification, we consider this approach as the reference for the problem of isosurfacing code verification.

2.1.4 Perspectives

Three-dimensional numerical simulation established itself as a necessary tool for knowledge discovery in many fields of science. It enables to evaluate, improve and analyze theoretical models especially when experimental validation is made difficult for financial, technical or legal reasons. In industry, simulation is ubiquitous in the modeling process of a product.

Traditionally, such simulations are run on High-Performance Computing (HPC) resources while their output (typically a scalar field representing a simulated quantity at a given time-step) is transfered on a remote work station for post-processing purposes: visualization and analysis.

This overall methodology turns out to be incompatible with the characteristics of the upcoming generation of super-computers (expected around 2018) with predicted computing performances at the ExaScale (10^{18} FLoating-point Operations Per Second, FLOPS), since:

1. it will come with unprecedented technical challenges that cannot be addressed by simply extending existing frameworks (Har12) and which will impose new constraints on data analysis algorithms;

2. it will also enable radically novel simulation usages (DA13) which will result in novel types of data to analyze.

These perspectives are discussed with further details in chapter 8.

In-Situ Data Analysis

Current estimations (Har12) expect an increasing imbalance between predicted data throughputs (10^{12} bytes/s (DA13)) and persistent storage (10^{10} bytes/s) or global network bandwidths (most HPC users are located off-site). In this scenario, traditional off-line post-processing is no longer an option given this increasing bottleneck.

Therefore, it is necessary to move data analysis algorithms as close as possible to their input source (the numerical simulation algorithms) in order to simply avoid this bottleneck. In particular, to minimize the usage of persistent storage, it is necessary that analysis algorithms run on the same hardware as the simulation and that they run during the simulation process.

Topological data analysis demonstrated its robustness, genericity and practical interest in various applications. However, the above requirements represent major theoretical challenges for Topological Data Analysis, that I will attempt to address in the next few years:

- *Parallel and Distributed Algorithms:* To make an optimal usage of HPC hardware, analysis algorithms should be able to run in parallel in a shared as well as in a distributed memory model. This is a radical challenge for the computation of Topological Abstractions, since its robustness precisely come from the global consistency of its algorithms. In particular, these rely on a global analysis of the data, requiring global low-level operations and data-structures (sort, breadth-first searches, union-find data-structures). While the parallelism of the latter operations and data-structures have been studied, existing solutions for these low level operations do not guarantee efficient and scalable topological data analysis algorithms, which have to be completely reviewed. I will attempt to address this problem.

- *Interruptible Algorithms:* In the Exascale scenario, the persistent storage of each time step of a solution will no longer be possible. Therefore, it is necessary to develop analysis algorithms capable of processing a time-step while it remains in the simulation code's memory. This means that analysis algorithms should be *interruptible*: they should be able to be stopped at any time (when a time-step is deleted from memory) while still providing a globally consistent approximation of the output (that can be used for further processing). A direction to achieve this effect is to design *coarse-to-fine* algorithms. This represents a major theoretical challenge for Topological Data Analysis, whose algorithms have been traditionally thought in a *fine-to-coarse* scheme (topological abstractions are first computed exactly and then simplified through mechanisms such as persistent homology). This issue raises several algorithmic challenges as well as theoretical questions (for instance, in a coarse-to-fine model, what is the relation between the computed coarse-to-fine hierarchy and the fine-to-coarse hierarchy provided by persistence simplification?). I will attempt to address both problems in the next few years.

Preliminary results I am currently working with the Master student Charles Gueunet and Pierre Fortin (assistant professor at the Scientific Computing department of Sorbonne Universités UPMC, with strong HPC expertise) on designing efficient and scalable algorithms for the Contour Tree computation, simplification and data segmentation for a shared memory parallel model. Our preliminary results in this topic are promising as they seem to indicate that a virtually linear scaling is expectable for

large-scale data-sets (which would outperform previous attempts of the state-of-the-art).

Starting in early 2016, Charles will start a Ph.D. thesis under my supervision in the framework of an industrial partnership with Kitware S.A.S., the French subsidiary of Kitware Inc. (worldwide leader in open-source visualization software). During this Ph.D. (2016-2019), we will address each of the problems mentioned above for the simple case of the Contour Tree. More advanced topological abstractions will be considered later on.

Analysis of Function Spaces

The HPC performance improvements observed over the last years allow new simulation usages that will become a standard at the ExaScale. This new usages raise new important problems for visualization and data analysis:

- *Multi-physics Simulations:* Given the recent HPC performances, it becomes now possible to model complex macroscopic processes by jointly simulating multiple physical phenomena (for instance: joint thermodynamic, computational fluid dynamics and material resistance simulations). In this context, a given simulation generates a *family* of scalar functions that represent drastically (and physically) different quantities (temperature, flow vorticity, material stress, etc.). This variability leads concretely to scalar fields having drastically different range values and dynamics. The joint analysis of several scalar fields is therefore a major challenge that needs to be addressed to identify and quantify possible geometrical correlations between quantities. However, traditionally, Topological Data Analysis only deals with the analysis of one scalar function defined on a single geometry. Therefore, in this topic, I will extend the concepts of Topological Data Analysis to *multi-variate* scalar functions. This effort will be accompanied with the design of algorithms that are efficient in practice for the construction and simplification of these generalized topological abstractions, and their exploitation in specific application problems.

  Preliminary results In the context of a collaboration with the University of Leeds, we are currently working on the design of an algorithm with efficient practical performances for the problem of Reeb space computation (EHP08). This construction generalizes the notion of Reeb graph to multivariate scalar functions. While an al-

gorithm has been described for its approximation (CD14), no algorithm with efficient practical behavior has been documented for its exact computation and simplification. We addressed this problem and I am currently implementing this approach. In the process of defining this algorithm, we needed to introduce a novel construction called *Fiber Surfaces*, which are pre-images of curves through bivariate functions. Surprisingly we discovered that this novel construction was somehow implicitly known by the volume rendering community for multi-dimensional transfer function definition. However, this community could only visualize a volume rendering of these fiber surfaces and no algorithm was documented to extract them geometrically. We therefore presented a simple approach to compute fiber surfaces and showed the applicative interests of such constructions for data segmentation purposes in various application fields (CGT*15). However this algorithm was slow in practice for large data-sets and was only approximate. Therefore, we introduce a second algorithm for the exact computation of fiber surfaces as well as several acceleration mechanisms (generalized from isosurface extraction acceleration) which enabled an interactive exploration of the space of fiber surfaces (KTCG15). We consider this latter algorithm as the reference for the problem of efficient and exact fiber surface computation in bivariate scalar fields. In this topic, I will therefore continue this research effort by generalizing other topological abstractions and deriving other application driven algorithms capable of processing multivariate data.

- *Uncertain Simulations:*

  A physical model is often dictated by a number of parameters (initial conditions, boundary conditions, etc.). Given the recent HPC advances, the fine sampling of this parameter space becomes feasible (yielding one simulation output per combination of parameters). This process, called parameter study, is central to the understanding of the uncertainty that accompanies any physical process. For instance, it enables to identify parameter ranges for an efficient and safe functioning of a complex system. This type of simulation also generates a *family* of scalar fields, that model an uncertain process. Analyzing this family as a whole to identify, extract and understand the conditions of appearance of features of interest is a major upcoming challenge in visualization and analysis.

From a theoretical point of view, the topological analysis of multivariate fields seems to have an applicative interest only when the dimension of the range is lower than the dimension of the domain (typically three). Beyond, another direction should be considered. Therefore, the topological analysis of multivariate fields seems of little importance for the processing of uncertain data. Instead, in this topic, I will address the problem of generalizing the constructions of Topological Data Analysis to uncertain scalar fields (that map each point of the domain to a random variable).

Preliminary results In the context of an exploratory project for which I was the principal investigator (called "UnTopoVis", national funding, Digiteo), I introduced the first non-local, combinatorial characterization of critical points and their global relation in 2D uncertain scalar fields (GST14). The characterization is based on the analysis of the support of the probability density functions of the input random variables. Given two scalar fields representing reliable estimations of the bounds of this support, our strategy identifies mandatory critical points: spatial regions and function ranges where critical points have to occur in any realization of the input uncertain data. The algorithm provides a global pairing scheme for mandatory critical points which is used to construct mandatory join and split trees. These trees enable a visual exploration of the common topological structure of all possible realizations of the uncertain data. To allow multi-scale visualization, we introduce a simplification scheme for mandatory critical point pairs revealing the most dominant features. Our technique is purely combinatorial and handles parametric distribution models and ensemble data. It does not depend on any computational parameter and does not suffer from numerical inaccuracy or global inconsistency. The algorithm exploits ideas of the established join/split tree computation. It is therefore simple to implement, and its time complexity is output-sensitive. Experiments demonstrated the accuracy and usefulness of our method on synthetic and real-world uncertain data-sets. Thanks to its robustness and multi-scale nature, we consider this approach as the reference algorithm for the problem of mandatory critical point extraction in 2D uncertain scalar fields. Despite this strong result, this first attempt at extending Topological Data Analysis to uncertain data raised even more questions than answers. In particular, de-

spite their strong applicative interest, the topological features that are common to all realizations of an uncertain process (i.e. *that have a probability of appearance of 1*) only constitute a sub-set of the features users are interested in. From a theoretical point of view, a natural question that arises is "What about the critical points with a probability of appearance lower than 1?". This question has strong applicative implications. Often, the phenomena under investigation can reveal distinct regimes and it is important to understand the probability of appearance of these regimes as well as the conditions (sets of parameters) for their appearance. This is one of the goals of the AVIDO research project (started in October 2015), for which I am a local investigator and for which I will hire an engineer and a post-doctoral researcher.

2.1.5 Development Platform

Since early 2014, I develop a C++ visualization platform that wraps around the open-source platform ParaView, codenamed *WTFIT*, that is specialized for the development of efficient Topological Data Analysis algorithms. Its core algorithms are based on a previous platform (based on OpenGL only) that I started to write in 2010. Its purpose is fourfold:

1. Enable an easy and rapid writing of advanced and efficient topological data analysis programs. In particular, since it relies on ParaView, the developer only needs to focus on the writing of the core analysis algorithm, without having to deal with IO, rendering or interaction. Moreover, thanks to the ParaView pipeline hierarchy, the developer can easily combine its data analysis algorithms with any of the routines already implemented in ParaView.

2. Centralize all the developments of my students and I into one coherent software program.

3. Facilitate the technological transfer of my algorithms to collaboration partners. Since ParaView is one of the most popular visualization platform, it takes only little effort for users already familiar with ParaView to interact with my algorithms.

4. Facilitate code dissemination. Since ParaView and VTK are open-source, it is easy for other researchers to compile and run the modules of WTFIT for reproduction purposes.

Publication Type	Number
Edited books	1
Peer-reviewed journals	17
Peer-reviewed conferences	10
Peer-reviewed book chapters	2
Invited conference papers	2

Table 2.1 – *Publication summary.*

Ever since, all my students develop their research projects in this platform as individual modules. I wrote an extensive documentation as well as a detailed tutorial (including examples) that I ask all my students to go through during the first week of their internship or Ph.D. thesis, as a training process.

As much as possible, each time a paper will be published, the corresponding module will be released in open-source together with the paper.

To my understanding, and this is one of the core lessons that I learned at the Scientific Computing and Imaging Institute, research advances have to be accompanied with usable and efficient software to advertise the work and more importantly, to make dissemination and transfer really happen. Moreover, this embraces my personal view as a computer scientist since software is in the end the long term goal of our research.

2.1.6 Publications

Pre-prints of the following publications can be downloaded from my web page: http://lip6.fr/Julien.Tierny.

Ph.D. Thesis

1. **Julien Tierny**, *"Reeb graph based 3D shape modeling and applications"*, Ph.D. thesis. *Committee*: S. Tison (President), A. Baskurt (Reviewer), B. Lévy (Reviewer), C. Labit (Committee), A. Srivastava (Committee), M. Daoudi (Advisor), J.P. Vandeborre (Co-advisor). Lille1 University, October 2008.

Edited books

2. Valerio Pascucci, Xavier Tricoche, Hans Hagen, and **Julien Tierny**, *"Topological Methods in Data Analysis and Visualization: Theory, Algorithms and Applications"*, Springer, 2011 (ISBN: 978-3-642-15013-5).

Peer-reviewed journals

3. Hamish Carr, Zhao Geng, **Julien Tierny**, Amit Chattopadhyay, Aaron Knoll, *"Fiber Surfaces: Generalizing Isosurfaces to Bivariate Data"*, Computer Graphics Forum, Proc. of EuroVis 2015.

4. Sujin Philip, Brian Summa, **Julien Tierny**, Peer-Timo Bremer, Valerio Pascucci, *"Distributed Seams for Gigapixel Panoramas"*, IEEE Transactions on Visualization and Computer Graphics, 2015.

5. Attila Gyulassy, David Guenther, Joshua Levine, **Julien Tierny**, Valerio Pascucci, *"Conforming Morse-Smale Complexes"*, IEEE Transactions on Visualization and Computer Graphics, Proc. of IEEE VIS 2014.

6. David Guenther, Roberto Álvarez Boto, Julia Contreras Garcia, Jean-Philip Piquemal, **Julien Tierny**, *"Characterizing Molecular Interactions in Chemical Systems"*, IEEE Transactions on Visualization and Computer Graphics, Proc. of IEEE VIS 2014.

7. David Guenther, Joseph Salmon, **Julien Tierny**, *"Mandatory Critical Points of 2D Uncertain Scalar Fields"*, Computer Graphics Forum, Proc. of EuroVis 2014.

8. Jean-Marc Thiery, **Julien Tierny**, Tamy Boubekeur, *"Jacobians and Hessians of Mean Value Coordinates for Closed Triangular Meshes"*, The Visual Computer, 2013.

9. **Julien Tierny** and Valerio Pascucci, *"Generalized Topological Simplification of Scalar Fields on Surfaces"*, IEEE Transactions on Visualization and Computer Graphics, Proc. of IEEE VIS 2012.

10. **Julien Tierny**, Joel Daniels II, Luis Gustavo Nonato, Valerio Pascucci and Claudio Silva, *"Interactive Quadrangulation with Reeb Atlases and Connectivity Textures"*, IEEE Transactions on Visualization and Computer Graphics, 2012.

11. Brian Summa, **Julien Tierny** and Valerio Pascucci, *"Panorama Weaving: Fast and Flexible Seam Processing"*, ACM Transactions on Graphics, Proc. of ACM SIGGRAPH 2012.

12. Jean-Marc Thiery, **Julien Tierny** and Tamy Boubekeur, *"CageR: Cage-based Reverse Engineering of Animated 3D Shapes"*, Computer Graphics Forum, 2012.

13. Tiago Etiene, Luis Gustavo Nonato, Carlos Scheidegger, **Julien Tierny**, Tom Peters, Valerio Pascucci, Mike Kirby and Claudio Silva, *"Topology Verification for Isosurface Extraction"*, IEEE Transactions on Visualization and Computer Graphics, 2012.

14. Jean-Marc Thiery, Bert Buchholz, **Julien Tierny** and Tamy Boubekeur, *"Analytic Curve Skeletons for 3D Surface Modeling and Processing"*, Computer Graphics Forum, Proc. of Pacific Graphics 2012.

15. **Julien Tierny**, Joel Daniels II, Luis Gustavo Nonato, Valerio Pascucci and Claudio Silva, *"Inspired Quadrangulation"*, Computer Aided Design, Proc. of ACM Solid and Physical Modeling 2011.

16. Peer-Timo Bremer, Gunther Weber, **Julien Tierny**, Valerio Pascucci, Marc Day and John Bell, *"Interactive Exploration and Analysis of Large Scale Simulations Using Topology-based Data Segmentation"*, IEEE Transactions on Visualization and Computer Graphics, 2010.

17. **Julien Tierny**, Attila Gyulassy, Eddie Simon and Valerio Pascucci, *"Loop surgery for volumetric meshes: Reeb graphs reduced to contour trees"*, IEEE Transactions on Visualization and Computer Graphics, Proc. of IEEE VIS 2009.

18. **Julien Tierny**, Jean-Philippe Vandeborre and Mohamed Daoudi, *"Partial 3D shape retrieval by Reeb pattern unfolding"*, Computer Graphics Forum, 2009.

19. **Julien Tierny**, Jean-Philippe Vandeborre and Mohamed Daoudi, *"Enhancing 3D mesh topological skeletons with discrete contour constrictions"*, The Visual Computer, 2008.

Peer-reviewed conferences

20. Sujin Philip, Brian Summa, **Julien Tierny**, Peer-Timo Bremer, Valerio Pascucci, *"Scalable Seams for Gigapixel Panoramas"*, Eurographics Symposium on Parallel Graphics and Visualization, 2013.

21. Mariem Gargouri, **Julien Tierny**, Erwan Jolivet, Philippe Petit, Elsa Angelini, *"Accurate and robust shape descriptors for the identification of rib cage structures in CT-images with Random Forests"*, IEEE International Symposium on Biomedical Imaging, 2013.

22. Jean-Christophe Michelin, **Julien Tierny**, Florence Tupin, Clément Mallet, Nicolas Paparoditis, *"Quality Evaluation of 3D City Building Models with Automatic Error Diagnosis"*, Proc. of ISPRS Conference on SSG, 2013.

23. Emanuele Santos, **Julien Tierny**, Ayla Khan, Brad Grimm, Lauro Lins, Juliana Freire, Valerio Pascucci, Claudio Silva, Scott Klasky, Roselyne Barreto, Norbert Podhorszki, *"Enabling Advanced Visualization Tools in a Web-Based Simulation Monitoring System"*, IEEE International Conference on eScience, 2009.

24. Peer-Timo Bremer, Gunther Weber, **Julien Tierny**, Valerio Pascucci, Marcus Day, John Bell, *"A Topological Framework for the Interactive Exploration of Large Scale Turbulent Combustion"*, IEEE International Conference on eScience, 2009.

25. **Julien Tierny**, Jean-Philippe Vandeborre and Mohamed Daoudi, *"Fast and precise kinematic skeleton extraction of 3D dynamic meshes"*, IEEE International Conference on Pattern Recognition, 2008,

26. **Julien Tierny**, Jean-Philippe Vandeborre and Mohamed Daoudi, *"Reeb chart unfolding based 3D shape signatures"*, Eurographics 2007 (short papers).

27. **Julien Tierny**, Jean-Philippe Vandeborre and Mohamed Daoudi, *"Topology driven 3D mesh hierarchical segmentation"*, IEEE Shape Modeling International 2007 (short papers).

28. **Julien Tierny**, Jean-Philippe Vandeborre and Mohamed Daoudi, *"3D mesh skeleton extraction using topological and geometrical analyses"*, Pacific Graphics 2006.

29. **Julien Tierny**, Jean-Philippe Vandeborre and Mohamed Daoudi, *"Invariant high level Reeb graphs of 3D polygonal meshes"*, IEEE 3DPVT, 2006.

Peer-reviewed book chapters

30. **Julien Tierny**, David Guenther, and Valerio Pascucci, *"Optimal General Simplification of Scalar Fields on Surfaces"*, chapter of *"Topological and Statistical Methods for Complex Data"*, Springer, 2014 (ISBN: 978-3-662-44899-1).

31. Stefano Berretti, Mohamed Daoudi, Alberto Del Bimbo, Tarik Filali Ansary, Pietro Pala, **Julien Tierny** and Jean-Philippe Vandeborre, *"3D object indexing"*, chapter of *"3D object processing: compression, indexing and watermarking"*, Ed. Wiley, 2008 (ISBN: 978-0-470-06542-6).

Invited conference papers

32. **Julien Tierny**, Jean-Philippe Vandeborre and Mohamed Daoudi, *"Geometry flavored topological skeletons: applications to shape handling, understanding and retrieval"*, Second DELOS Conference, 2007.

33. Mohamed Daoudi, Tarik Filali-Ansary, **Julien Tierny** and Jean-Philippe Vandeborre, *"3D mesh models: view-based indexing and structural analysis"*, First DELOS Conference, 2007, Lecture Notes in Computer Science.

2.1.7 Research grants

Since 2008, my research has been partially funded by the following research grants:

1. 2008-2009: Fulbright fellowship (US Department of State);
2. 2008-2009: Lavoisier fellowship (French Ministry for Foreign Affairs);
3. 2013-2014: "Uncertain TopoVis" project (2012-063D, national funding, Digiteo grant), principal investigator;
4. 2013-2017: "CrABEx" project (ANR-13-CORD-0013, national funding, ANR grant), local investigator;

5. 2015-2018: "AVIDO" project (national funding, BPI grant), local investigator;
6. 2016-2019: "In-Situ Topological Data Analysis" project (partnership with Kitware S.A.S.), principal investigator.

I also contributed to the following research projects:

1. 2011-2015: "Car passenger modeling with medical imaging" project (partnership with Renault);
2. 2013-present: "CalSimLab" project (ANR-11-LABX-0037-01);
3. 2014-present: "Riemannian surface visualization" project (DGA DT-SCAT-DA-IDF DF1300034MNRBC).

2.1.8 Students

Since my Ph.D. thesis defense, I had the opportunity to advise the following students:

Master students In 2014, I was the main advisor for the master internships (4 months) of Kenny Peou and Chantal Ding and I co-advised in 2015 the internships (6 months) of Matthew Henry, Charles Gueunet and Guillaume Favelier. Charles and Guillaume will continue to work under my supervision after their internships as a Ph.D. student and a software engineer respectively.

Ph.D. students I co-advised Mariem Gargouri's Ph.D. thesis (defended in June 2015) in the framework of the project "Car passenger modeling with medical imaging". I collaborate with the Ph.D. student Roberto Alvarez Boto (defense expected in 2016) in the framework of the project "CalSimLab". I am the main advisor of the Ph.D. thesis of Ana Vintescu (to be defended in 2017) in the framework of the project "CrABEx" and the main advisor of the Ph.D. thesis of Charles Gueunet (starting in early 2016). I also worked in close collaboration with Jean-Marc Thiery at Telecom ParisTech (defense in 2012, now a post-doctoral researcher at TU Delft, Netherlands) and Brian Summa at the University of Utah (defense in 2013, now an assistant professor at Tulane University, USA).

Post-docs I was the main advisor of the post-doctoral researcher David Guenther (2013-2014, now an engineer with Sirona Dental Systems) in the framework of the "Uncertain TopoVis" project.

2.1.9 Service

In addition to my research activity, I served my scientific community in the following ways:

Keynote speaker and invited talks I was a keynote speaker for the IEEE Shape Modeling International 2015 conference. I also gave invited talks in other events (CEA Uncertainty Forum 2014, Franco-Romanian applied mathematics congress 2014, AC3D Worskshop 2014) and universities (Tulane University, University of Leeds, Max Planck Institut fur Informatik, Clemson University and other French universities).

Event organizer I was the co-chair of the poster track of the IEEE Large Data Analysis and Visualization 2014 symposium. I was the organizer of the French workshop on Visualization in 2012 in Telecom ParisTech (about 80 attendees).

International Program Committees I served on the following international program committees:

- EuroVis 2015-2016 (full papers);
- IEEE Shape Modeling International 2015 (full papers);
- TopoInVis 2015 (full papers);
- EuroVis 2013-2014 (short papers);
- EuroGraphics 2012-2013 (short papers).

Reviews In addition to my reviewing service as a program committee member, I reviewed papers for the following journals and conferences (around 60 papers overall):

- *Journals:* IEEE Transactions on Visualization and Computer Graphics, Computer Graphics Forum, Computer Aided-Design, Computer-Aided Geometric Design, IEEE Transactions on Image Processing, International Journal of Computer Vision, Theoretical Computer Science, Image and Vision Computing;
- *Conferences:* IEEE VIS (2009, 2012-2015), EuroVis (2009, 2013-2016), ACM SIGGRAPH (2012-2013, 2015), ACM SIGGRAPH ASIA (2015), Eurographics (2009-2012, 2015-2016), Pacific Graphics (2011), ACM

Solid and Physical Modeling (2008), IEEE Shape Modeling International (2008, 2015), High Performance Graphics (2013), TopoInVis (2013, 2015), IEEE SIBGRAPI (2009), IEEE ICME (2007-2008).

Ph.D. committees I served as a jury member on the Ph.D. committees of the following students:

- Esma Elghoul (INRIA - Rocquencourt), 2014;
- Maxime Belperin (LIRIS), 2013;
- Rachid El Khoury (LIFL), 2013;
- Bertrand Pellenard (INRIA - Sophia Antipolis), 2012;
- Romain Arcila (LIRIS), 2011.

Fellowship committees I had the opportunity to serve at the Franco-American Commission as a committee member for the Fulbright fellowship (2011-2012).

2.2 Teaching activity

During my Ph.D. thesis, I was a teaching assistant (*"Moniteur"*) within the Computer Science department of Lille 1 University. Ever since, I continued to teach regularly since I consider teaching as a mandatory component of research dissemination. Moreover, I like to think that my teaching contributes to some extent to the development of a visualization community in France, to form skilled students both for the industry and the academia. Overall, I currently teach around 80 hours per year. All of my teaching material can be found on the following web-page: http://lip6.fr/Julien.Tierny/teaching.html.

2.2.1 Main classes

Since 2013, I am the head instructor of the scientific visualization class at ENSTA ParisTech (24 hours, Master-2 level) and at the University of Versailles (32 hours, Master-1 level). All of the teaching material for this class (lectures and exercises) is available on a dedicated web-page: http://lip6.fr/Julien.Tierny/visualizationClass.html. In particular, its online exercises are by far the most visited pages of my website (with more than 14,000 hits over the last year). I also contribute to the Computer

Graphics class at Telecom ParisTech and at Sorbonne Universités UPMC (with visualization and geometry processing lectures). Last year, I was able to recruit two students following my classes for their research internship.

Starting in 2016, I should also contribute to the visualization class of the computer science and applied mathematics program of the engineer school PolyTech Paris.

2.2.2 Past classes

In 2014, I contributed to the international summer school on scientific visualization organized by the Institute for Computing and Simulation (ICS) of Sorbonne Universités UPMC.

From 2008 to 2010, I regularly gave lectures at the University of Utah in the scientific visualization class (volume rendering and topological data analyis for undergraduate attendees) and the computational topology class (simplicial complexes, delaunay complexes and persistent homology for graduate attendees).

From 2005 to 2008, I gave several lectures and exercise sessions as a teaching assistant (*"Moniteur"*) at the Computer Science department of Lille1 University (64 hours per year).

Notations

$\mathbb{X}$	Topological space
$\partial\mathbb{X}$	Boundary of a topological space
$\mathbb{M}$	Manifold
$\mathbb{R}^d$	Euclidean space of dimension d
$\sigma,\ \tau$	d-simplex, face of a d-simplex
$v,\ e,\ t,\ T$	Vertex, edge, triangle and tetrahedron
$Lk(\sigma),\ St(\sigma)$	Link and star of a simplex
$Lk_d(\sigma),\ St_d(\sigma)$	d-simplices of the link and the star of a simplex
$\mathcal{K}$	Simplicial complex
$\mathcal{T}$	Triangulation
$\mathcal{M}$	Piecewise linear manifold
β_i	i-th Betti number
χ	Euler characteristic
α_i	i^{th} barycentric coordinates of a point p relatively to a simplex σ
$f : \mathcal{T} \to \mathbb{R}$	Piecewise linear scalar field
∇f	Gradient of a PL scalar field f
$Lk^-(\sigma),\ Lk^+(\sigma)$	Lower and upper link of σ relatively to f
$o(v)$	Memory position offset of the vertex v
$\mathcal{L}^-(i),\ \mathcal{L}^+(i)$	Sub- and sur-level set of the isovalue i relatively to f
$\mathcal{D}(f)$	Persistence diagram of f
$\mathcal{C}(f)$	Persistence curve of f
$\mathcal{R}(f)$	Reeb graph of f
$l(\mathcal{R}(f))$	Number of loops of $\mathcal{R}(f)$
$\mathcal{T}(f)$	Contour tree of f
$\mathcal{J}(f),\ \mathcal{S}(f)$	Join and split trees of f
$\mathcal{MS}(f)$	Morse-Smale complex of f

Background

3

Contents

This chapter introduces all the theoretical preliminaries required for the reading of the rest of the manuscript. First, the input data representation is formalized. Second, some of the core concepts of Topological Data Analysis are presented, including critical points, notions of Persistent Homology, Reeb graphs and Morse-Smale Complexes. Finally, a brief review of the state-of-the-art algorithms is presented.

For the reader's convenience, the most important definitions and properties are highlighted with boxes.

For further readings, I refer the reader to the excellent introduction to Computational Topology by Edelsbrunner and Harer (EH09).

3.1 Data representation

In scientific visualization, scalar data is in general defined on an input geometrical object (hereafter named *"Domain"*). It is represented by a finite set of sample values, continuously extended in space to the entirety of the domain thanks to an interpolant. In the following, I first formalize a generic domain representation. Next, I formalize a representation of the scalar data on this object (hereafter termed *"Range"*).

3.1.1 Domain representation

In the following, I formalize a generic domain representation. This notion is introduced constructively. At the end of the subsection, I further describe topological notions relative to this domain representation that will be used in the remainder of the manuscript.

Preliminary notions

Definition 1 (Topology) *A topology on a set $\mathbb{X}$ is a collection T of subsets of $\mathbb{X}$ having the following properties:*

- *The sets $\emptyset$ and $\mathbb{X}$ are in T;*
- *The union of any sub-collection of T is in T;*
- *The intersection of a finite sub-collection of T is in T.*

Definition 2 (Topological space) *A set $\mathbb{X}$ for which a topology T is defined is called a* topological space.

For example, the space of real numbers $\mathbb{R}$ is a topological space.

Definition 3 (Open set) *A subset $\mathbb{A} \subset \mathbb{X}$ of the topological space $\mathbb{X}$ is an* open *set of $\mathbb{X}$ if it belongs to T.*

Definition 4 (Closed set) *A subset $\mathbb{B} \subset \mathbb{X}$ of the topological space $\mathbb{X}$ is a* closed *set of $\mathbb{X}$ if its complement $\mathbb{X} - \mathbb{B}$ is open.*

Intuitively, open sets are subsets of topological spaces which do not contain their boundaries. For example, considering the space of real numbers $\mathbb{R}$, $(-\infty, 0) \cup (1, +\infty)$ and $[0, 1]$ are complements and respectively open and closed sets.

Property 1 (Open sets)

- *The set $\emptyset$ is open;*

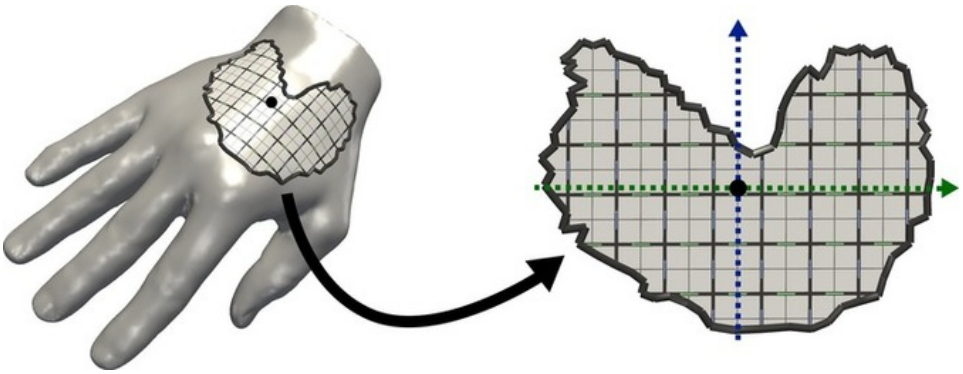


Figure 3.1 – *Example of* 2*-manifold: any point of the surface (left, black dot) has an open neighborhood (textured chart) that is homeomorphic to an open Euclidean* 2*-ball (that can be* unfolded *to the plane, right).*

- *The union of any number of open sets is open;*
- *The intersection of a finite number of open sets is open.*

These properties follow from the definition of topology.

Definition 5 (Covering) *A collection of subsets of a topological space* $\mathbb{X}$ *is a* covering *of* $\mathbb{X}$ *if the union of all its elements is equal to* $\mathbb{X}$*.*

Definition 6 (Compact topological space) *A topological space* $\mathbb{X}$ *is* compact *if every open covering of it contains a finite sub-collection that is also a covering of* $\mathbb{X}$*.*

Definition 7 (Function) *A* function $f : \mathbb{A} \to \mathbb{B}$ *associates each element of the topological space* $\mathbb{A}$ *with a unique element of the topological space* $\mathbb{B}$*.*

Definition 8 (Injection) *A function* $f : \mathbb{A} \to \mathbb{B}$ *is an* injection *if for each pair* $a_1, a_2 \in \mathbb{A}$ *such that* $a_1 \neq a_2$*,* $f(a_1) \neq f(a_2)$*.* f *is said to be* one-to-one*.*

Definition 9 (Bijection) *A function* $f : \mathbb{A} \to \mathbb{B}$ *is a* bijection *if for each element* $b \in \mathbb{B}$ *there is exactly one element* $a \in \mathbb{A}$ *such that* $f(a) = b$*.* f *is said to be bijective. It is also said to be* one-to-one *(injective) and* onto *(surjective).*

Definition 10 (Continuous function) *A function* $f : \mathbb{A} \to \mathbb{B}$ *is* continuous *if for each open subset* $\mathbb{C} \in \mathbb{B}$*, the set* $f^{-1}(\mathbb{C})$ *is an open subset of* $\mathbb{A}$*.*

Definition 11 (Homeomorphic spaces) *Two topological spaces* $\mathbb{A}$ *and* $\mathbb{B}$ *are* homeomorphic *if and only if there exists a continuous bijection* $f : \mathbb{A} \to \mathbb{B}$ *with a continuous inverse* $f^{-1} : \mathbb{B} \to \mathbb{A}$*.* f *is a* homeomorphism*.*

Definition 12

(Manifold) *A topological space* $\mathbb{M}$ *is a* d-manifold *if every element* $m \in \mathbb{M}$ *has an open neighborhood* $\mathbb{N}$ *homeomorphic to an open Euclidean* d*-ball.*

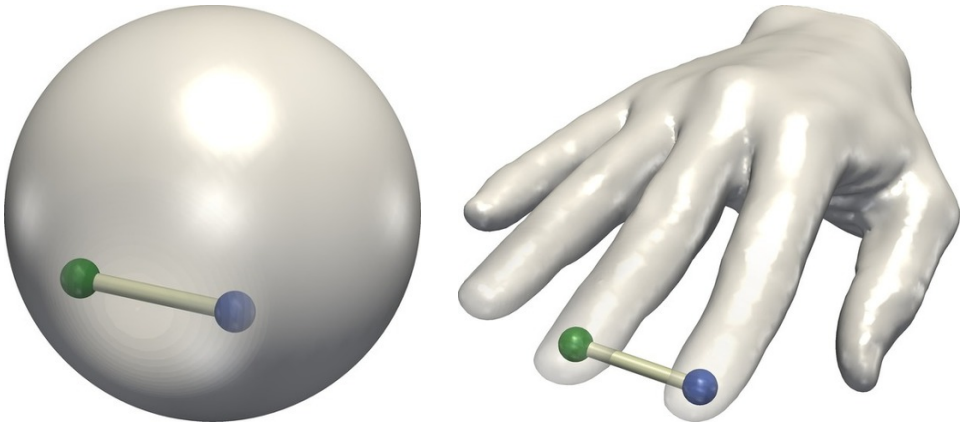


Figure 3.2 – *Examples of convex (left) and non-convex (right) 3-manifolds (volumes). On the left, any two points (green and blue spheres) can be linked by a line segment that belongs to the volume (white cylinder). This is not the case for the right volume.*

An intuitive description of a d-manifold is that of a curved space, which has locally the structure of an Euclidean space of dimension d, but which has a more complicated global structure (Euclidean spaces are therefore special cases of manifolds). Figure 3.1 illustrates this with the example of a 2-manifold (surface).

Domain formalization

In the following we formally introduce our domain representation as well as representations for connectivity information.

Definition 13 (Convex set) *A set* $\mathbb{C}$ *of an Euclidean space* $\mathbb{R}^n$ *of dimension* n *is* convex *if for any two points* x *and* y *of* $\mathbb{C}$ *and all* $t \in [0,1]$ *the point* $(1-t)x + ty$ *also belongs to* $\mathbb{C}$.

Intuitively, a convex set is a set such that any two points of the set can be linked by a line segment that belongs to the set, as illustrated with 3-manifolds (volumes) in Figure 3.2.

Definition 14 (Convex hull) *The* convex hull *of a set points* $\mathcal{P}$ *of an Euclidean space* $\mathbb{R}^n$ *is the unique minimal convex set containing all points of* $\mathcal{P}$.

Definition 15 (Simplex) *A* d-simplex *is the convex hull* σ *of* $d+1$ *affinely independent points of an Euclidean space* $\mathbb{R}^n$, *with* $0 \leq d \leq n$. d *is the dimension of* σ.

Definition 16 (Vertex) *A* vertex v *is a* 0*-simplex of* $\mathbb{R}^3$.

Definition 17 (Edge) *An* edge e *is a* 1*-simplex of* $\mathbb{R}^3$.

Definition 18 (Triangle) *A* triangle t *is a* 2*-simplex of* $\mathbb{R}^3$.

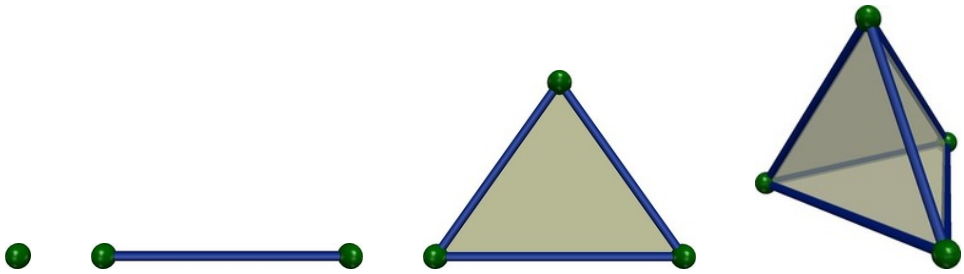


Figure 3.3 – *Illustrations of* 0 *(green),* 1 *(blue),* 2 *(white) and* 3*-simplices (transparent), from left to right, along with their faces.*

Definition 19 (Tetrahedron) *A* tetrahedron T *is a* 3*-simplex of* $\mathbb{R}^3$.

Definition 20 (Face) *A* face τ *of a* d*-simplex* σ *is the simplex defined by a non-empty subset of the* $d+1$ *points of* σ*, and is noted* $\tau \leq \sigma$*. We will note* τ_i *a face of dimension* i*.*

In summary, a d-simplex is the smallest combinatorial construction that can represent a neighborhood of a d-dimensional Euclidean space. As illustrated in Figure 3.3, it is composed of *faces*, that are themselves $(d-1)$, $(d-2)$, ..., and 0-simplices.

Definition 21 (Simplicial complex) *A* simplicial complex $\mathcal{K}$ *is a finite collection of non-empty simplices* $\{\sigma_i\}$*, such that every face* τ *of a simplex* σ_i *is also in* $\mathcal{K}$*, and any two simplices* σ_i *and* σ_j *intersect in a common face or not at all.*

Definition 22 (Star) *The* star *of a simplex* σ *of a simplicial complex* $\mathcal{K}$ *is the set of simplices of* $\mathcal{K}$ *that contain* σ*:* $St(\sigma) = \{\tau \in \mathcal{K}, \sigma \leq \tau\}$*. We will note* $St_d(\sigma)$ *the set of* d*-simplices of* $St(\sigma)$*.*

Definition 23

(Link) *The* link *of* σ *is the set of faces of the simplices of* $St(\sigma)$ *that are disjoint from* σ*:* $Lk(\sigma) = \{\tau \leq \Sigma, \Sigma \in St(\sigma), \tau \cap \sigma = \emptyset\}$*. We will note* $Lk_d(\sigma)$ *the set of* d*-simplices of* $Lk(\sigma)$*.*

In other words, the star of a simplex σ is the set of simplices having σ as a face, as illustrated Figure 3.4 (top). The notion of link is illustrated at the bottom of Figure 3.4.

Definition 24 (Underlying space) *The* underlying space *of a simplicial complex* $\mathcal{K}$ *is the union of its simplices* $|\mathcal{K}| = \cup_{\sigma \in \mathcal{K}} \sigma$*.*

Definition 25

(Triangulation) *The* triangulation $\mathcal{T}$ *of a topological space* $\mathbb{X}$ *is a simplicial complex* $\mathcal{K}$ *whose underlying space* $|\mathcal{K}|$ *is homeomorphic to* $\mathbb{X}$*.*

The notion of triangulation has been preferred here to other compet-

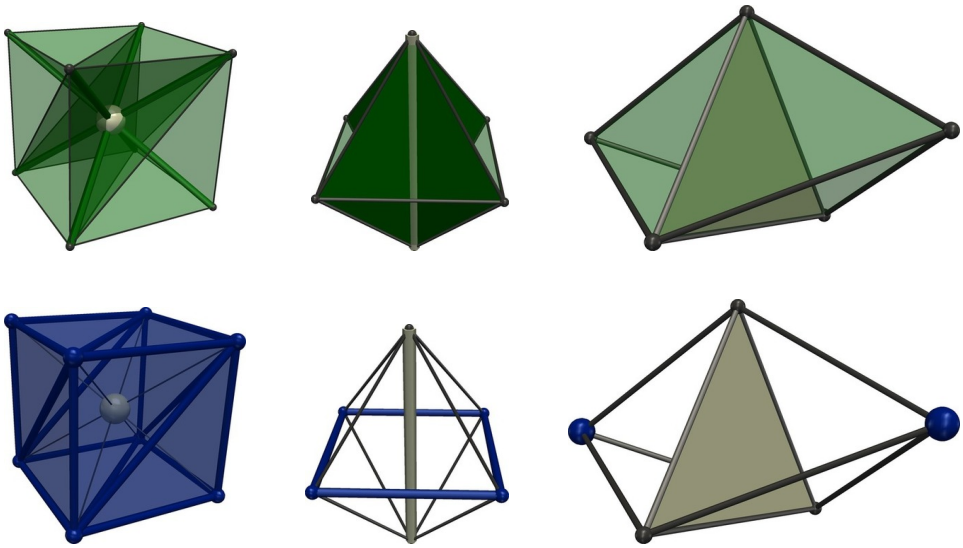


Figure 3.4 – *Illustrations of stars (green, top) and links (blue, bottom) for* 0*,* 1 *and* 2*-simplices (white, from left to right) of a* 3*-dimensional simplicial complex.*

ing representations for its practical genericity: any mesh representation (regular grid, unstructured grid, etc.) can be easily converted into a triangulation by subdividing each of its d-cells into valid d-simplices (having only $(d+1)$ linearly independent points), as illustrated in Figure 3.5 for the case of a regular grid. Also, note that for regular grids, the resulting triangulation can be implicitly encoded (i.e. adjacency relations can be retrieved on demand, without storage, thanks to the recurring subdivision pattern of the regular grid). Moreover, as detailed in the next subsection, triangulations can be accompanied with well-behaved interpolants, which facilitate reasoning and computation with scalar data.

As discussed further down this manuscript, for reasoning and robustness purposes, the following, more restrictive, notion is often preferred over triangulations.

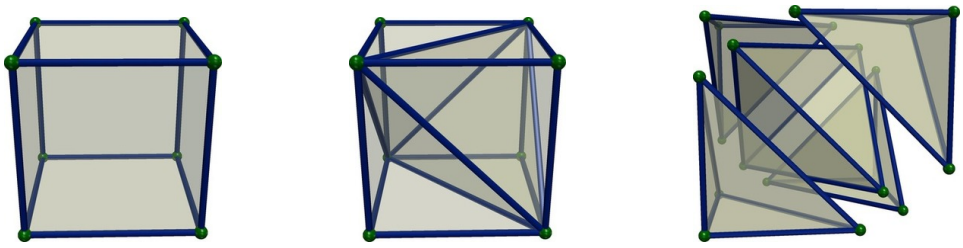


Figure 3.5 – *A* 3*-dimensional regular grid (left) can be easily converted into a triangulation by subdividing each of its voxels independently into* 5 *tetrahedra (center, right: exploded view). This subdivision can be implicitly encoded.*

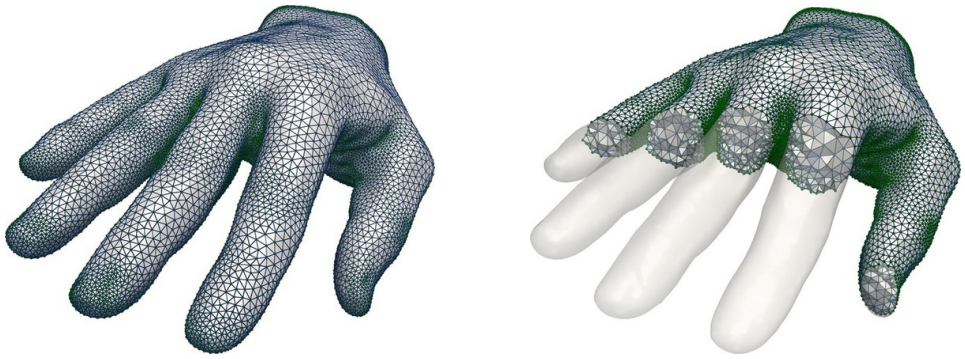


Figure 3.6 – *Example of PL 3-manifold (left, right: clipped view).*

Definition 26

(Piecewise Linear Manifold) *The triangulation of a manifold* $\mathbb{M}$ *is called a* piecewise linear manifold *and is noted* $\mathcal{M}$.

Therefore, a piecewise linear (PL) manifold is a combinatorial representation of a manifold that derives from the notion of triangulation, as illustrated in Figure 3.6. It can be efficiently represented in memory by storing for each dimension d, the list of d-simplices as well as their stars and links. In the remainder of this manuscript, we will consider PL-manifolds as our generic domain representations.

Topological invariants

In the following, I describe a few *topological invariants*: entities that do not change under continuous transformations of the domain (variations in point positions but no variation in connectivity). These notions are instrumental in Topological Data Analysis.

Definition 27 (Path) *A homeomorphism* $p : (a, b) \to \mathbb{C}$ *from an open interval* $(a, b) \subseteq \mathbb{R}$ *to a subset* $\mathbb{C}$ *of a topological space* $\mathbb{X}$ *is called a* path *on* $\mathbb{X}$ *between* $p(a)$ *and* $p(b)$.

Definition 28 (Connected topological space) *A topological space* $\mathbb{X}$ *is* connected *if for any two points of* $\mathbb{X}$ *there exists a path between them on* $\mathbb{X}$.

Definition 29 (Connected components) *The maximally connected subsets of a topological space* $\mathbb{X}$ *are called its* connected components.

Definition 30 (Homotopy) *A* homotopy *between two continuous functions* f *and* g *is a continuous function* $H : \mathbb{X} \times [0, 1] \to \mathbb{Y}$ *from the product of a topological space* $\mathbb{X}$ *with the closed unit interval to a topological space* $\mathbb{Y}$ *such that for each point* $x \in \mathbb{X}$, $H(x, 0) = f(x)$ *and* $H(x, 1) = g(x)$. *If there exists a homotopy between them,* f *and* g *are said to be* homotopic.

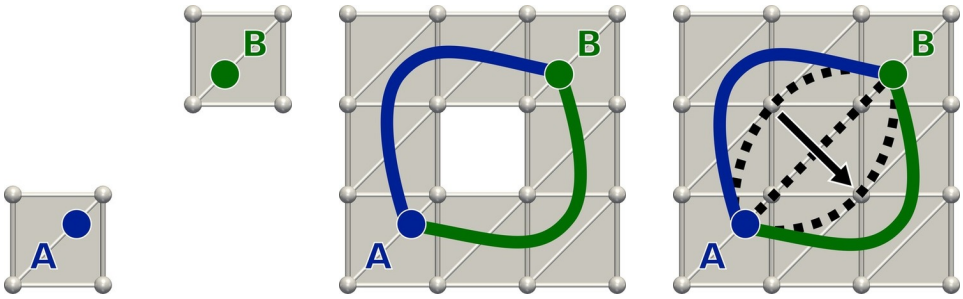


Figure 3.7 – *Examples of disconnected, connected and simply connected domains (from left to right).*

While homeomorphism deals with the matching between neighborhoods, homotopies additionally require that a continuous transformation exist between them, by considering neighborhoods as images of functions (the notion of homotopy is then refined to that of isotopy). Here, the second parameter of an homotopy can be seen as time in this continuous transformation process. For instance, a circle and a knot are homeomorphic but are not homotopic since the knot needs to be cut and stitched back to be turned into a circle, which is not a continuous transformation.

Definition 31

(Simply connected topological space) *A topological space* $\mathbb{X}$ *is* simply connected *if it is connected and if for any two points of* $\mathbb{X}$*, any two paths between them on* $\mathbb{X}$ *are homotopic.*

As illustrated in Figure 3.7, a domain is not simply connected if for any two points, any pair of paths between them cannot be continuously transformed into one another (black paths in Figure 3.7, right).

Definition 32 (Boundary) *The* boundary *of a topological space* $\mathbb{X}$*, noted* $\partial\mathbb{X}$*, is the complement in* $\mathbb{X}$ *of the subspace of* $\mathbb{X}$*, called the* interior *of* $\mathbb{X}$*, composed of all the elements* $x \in \mathbb{X}$ *such that* x *has an open neighborhood* $\mathbb{N}$*.*

Definition 33 (Boundary component) *A* boundary component *of a topological space* $\mathbb{X}$ *is a connected component of its boundary* $\partial\mathbb{X}$*.*

Definition 34 (p-chain) *A* p-chain *of a triangulation* $\mathcal{T}$ *of a topological space* $\mathbb{X}$ *is a formal sum (with modulo* 2 *coefficients) of* p*-simplices of* $\mathcal{T}$*.*

Definition 35 (p-cycle) *A* p-cycle *of a triangulation* $\mathcal{T}$ *of a topological space* $\mathbb{X}$ *is a* p*-chain with empty boundary.*

Definition 36 (Group of p-cycles) *The* group of p-cycles *of a triangulation* $\mathcal{T}$ *of a topological*

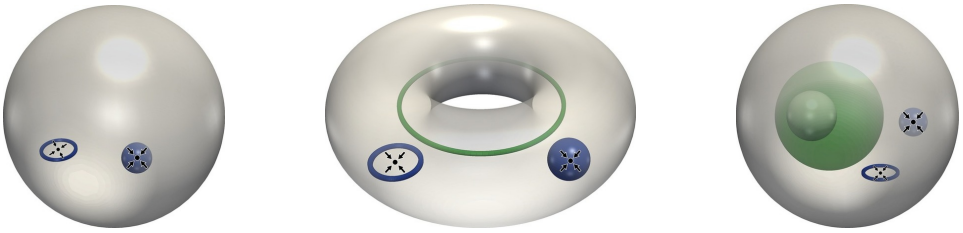


Figure 3.8 – *Examples of PL 3-manifolds with varying Betti numbers. From left to right: a 3-ball, a solid torus, a 3-ball with a void. From left to right, $(\beta_0, \beta_1, \beta_2)$ is equal to $(1,0,0)$, $(1,1,0)$, and $(1,0,1)$). Generators are displayed in green, while examples of non-generator p-cycles are displayed in blue.*

space $\mathbb{X}$ is the group of all p-cycles of $\mathcal{T}$, noted $Z_p(\mathcal{T})$, which forms a sub-group of all p-chains of $\mathcal{T}$.

Definition 37 (p-boundary) *A p-boundary of a triangulation $\mathcal{T}$ of a topological space $\mathbb{X}$ is the boundary of a $(p+1)$-chain.*

Property 2 (p-boundary) *A p-boundary is a p-cycle.*

Definition 38 (Group of p-boundaries) *The* group of p-boundaries *of a triangulation $\mathcal{T}$ of a topological space $\mathbb{X}$ is the group of all p-boundaries of $\mathcal{T}$, noted $B_p(\mathcal{T})$, which forms a sub-group of all p-cycles of $\mathcal{T}$.*

Definition 39 (Homology group) *The* pth homology group *of a triangulation $\mathcal{T}$ of a topological space $\mathbb{X}$ is its p^{th} cycle group modulo its p^{th} boundary group: $H_p(\mathcal{T}) = Z_p(\mathcal{T})/B_p(\mathcal{T})$.*

Intuitively, two p-cycles are said to be equivalent, or homologous, if they can be continuously transformed into each other (through formal sums with modulo 2 coefficients) without being collapsible to a point. Then, one can further group p-cycles into *classes* of equivalent p-cycles. Each class can be represented by a unique representative p-cycle that is called *generator* (and that is homologous to any other p-cycle of the class), as illustrated in Figure 3.8 with a green 1-cycle (center) and a green 2-cycle (right). Enumerating the number of generators of a homology group enables to introduce intuitive topological invariants called Betti numbers.

Definition 40

(Betti number) *The* pth Betti number *of a triangulation $\mathcal{T}$ of a topological space $\mathbb{X}$ is the rank of its p^{th} homology group: $\beta_p(\mathcal{T}) = rank(H_p(\mathcal{T}))$.*

In low dimensions, Betti numbers have a very concrete interpretation. For instance, for PL 3-manifolds, β_0 corresponds to the number of connected components, β_1 to the number of handles and β_2 to the number of

voids, as illustrated in Figure 3.8 (β_3 is equal to 0 for PL 3-manifolds with boundary, i.e. that can be embedded in $\mathbb{R}^3$).

Definition 41

(Euler characteristic) *The* Euler characteristic *of a triangulation $\mathcal{T}$ of a topological space $\mathbb{X}$ of dimension d, noted $\chi(\mathcal{T})$, is the alternating sum of its Betti numbers: $\chi(\mathcal{T}) = \sum_{i=0}^{i=d}(-1)^i\beta_i(\mathcal{T})$.*

Property 3 (Euler characteristic) *The* Euler characteristic *of a triangulation $\mathcal{T}$ of a topological space $\mathbb{X}$ of dimension d is also equal to the alternating sum of the number of its i-simplices: $\chi(\mathcal{T}) = \sum_{i=0}^{i=d}(-1)^i|\sigma_i|$.*

3.1.2 Range representation

In the following, I formalize a range representation based on the previously introduced domain representation. Additionally, I will introduce a few related geometrical constructions that will be instrumental to Topological Data Analysis.

Piecewise linear scalar fields

Definition 42

(Barycentric coordinates) *Let p be a point of $\mathbb{R}^n$ and σ a d-simplex. Let α_0, α_1, ..., α_d be a set of real coefficients such that $p = \sum_{i=0}^{i=d}\alpha_i\tau_0^i$ (where τ_0^i is the i^{th} zero dimensional face of σ) and such that $\sum_{i=0}^{i=d}\alpha_i = 1$. Such coefficients are called the* barycentric coordinates *of p relatively to σ.*

Property 4 (Barycentric coordinates) *The barycentric coordinates of p relative to σ are unique.*

Property 5 (Barycentric coordinates) *If and only if there exists an i for which $\alpha_i \notin [0,1]$, then p does not belong to σ, otherwise it does.*

Definition 43

(Piecewise Linear Scalar Field) *Let $\hat{f}$ be a function that maps the 0-simplices of a triangulation $\mathcal{T}$ to $\mathbb{R}$. Let $f : \mathcal{T} \to \mathbb{R}$ be the function linearly interpolated from $\hat{f}$ such that for any point p of a d-simplex σ of $\mathcal{T}$, we have: $f(p) = \sum_{i=0}^{i=d}\alpha_i\hat{f}(\tau_0^i)$ (where τ_0^i is the i^{th} zero dimensional face of σ). f is called a* piecewise linear (PL) scalar field.

Piecewise linear scalar fields will be our default representation for scalar data. Typically, the input data will then be given in the form of a triangulation with scalar values attached to its vertices ($\hat{f}$). The linear interpolation provided by the barycentric coordinates can be efficiently

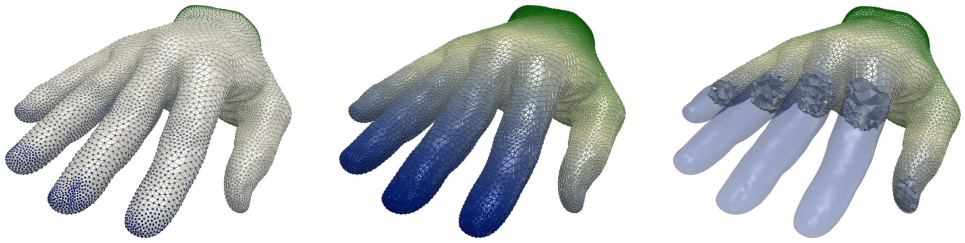


Figure 3.9 – *Example of PL scalar field f defined on a PL 3-manifold $\mathcal{M}$. From left to right: restriction $\hat{f}$ of f on the 0-simplices of $\mathcal{M}$, f (the color coding denotes the linear interpolation within each simplex), clipped view of f.*

computed on demand (on the CPU or the GPU, as illustrated in Figure 3.9) and has several nice properties that makes it well suited for combinatorial reasonings.

Property 6 *(Gradient of a Piecewise Linear Scalar Field) The* gradient ∇f *of a PL scalar field $f : \mathcal{T} \to \mathbb{R}$ is a curl free vector field that is piecewise constant (constant within each d-simplex of $\mathcal{T}$).*

This property has several implications that will be discussed in the following subsections.

Definition 44 (Lower Link) *The* lower link $Lk^-(\sigma)$ *(respectively the* upper link $Lk^+(\sigma)$*) of a d-simplex σ relatively to a PL scalar field f is the subset of the link $Lk(\sigma)$ such that each of its zero dimensional faces has a strictly lower (respectively higher) f value than those of σ.*

Given the above definition, it is often useful to disambiguate configurations of equality in f values between vertices (thus equalitiy configurations in $\hat{f}$). Therefore, $\hat{f}$ is often slightly perturbed with a mechanism inspired by simulation of simplicity (EM90) to turn $\hat{f}$ into an injective function. This can be achieved by adding to $\hat{f}$ a second function $\hat{g}$ that is injective (the result is then injective as well). Let $o(v)$ denote the position integer offset of the vertex v in memory. $o(v)$ is injective. Then, to turn $\hat{f}$ into an injective function, one needs to add to it $\epsilon o(v)$ where ϵ is an arbitrarily small real value. As the original simulation of simplicity, this mechanism can be implemented numerically (by choosing the smallest possible value for ϵ depending on the machine precision) or preferably symbolically by re-implementing the necessary predicates. For instance, to decide if a vertex v_0 is lower than a vertex v_1, one needs to test $\hat{f}(v_0) < \hat{f}(v_1)$ and, in case of equality, test $o(v_0) < o(v_1)$ to disambiguate. In the following, we will therefore consider that $\hat{f}$ is always injective in virtue of this mecha-

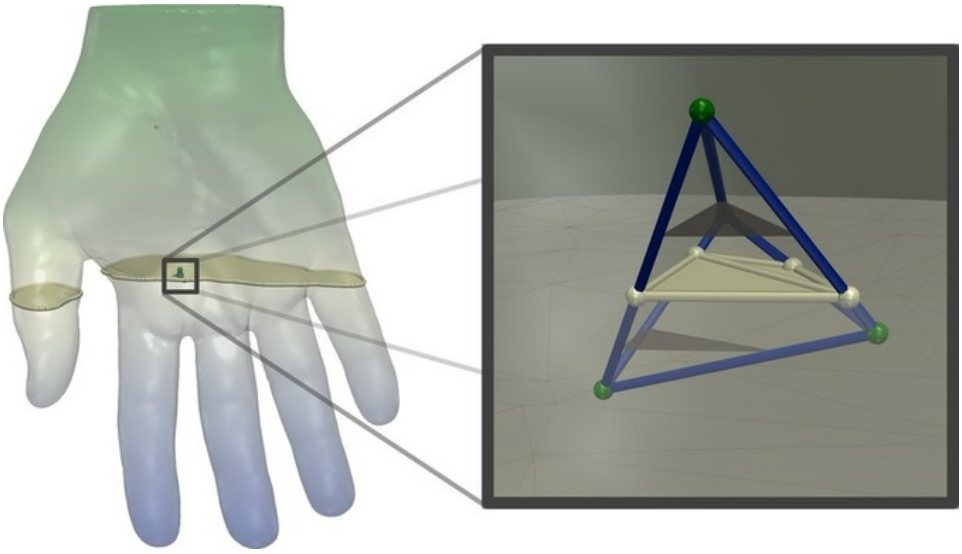


Figure 3.10 – *Example of level set (isosurface, left) of a PL scalar field defined on a PL 3-manifold. Right: restriction of the isosurface to a 3-simplex.*

nism. Therefore, no d-simplex of $\mathcal{T}$ collapses to a point of $\mathbb{R}$ through f for any non-zero d.

Related geometrical constructions

Based on our representation for scalar data on geometrical domains, I will now introduce a few geometrical constructions that will be instrumental in Topological Data Analysis.

Definition 45 (Sub-level set) *The* sub-level set $\mathcal{L}^-(i)$ *(respectively the* sur-level set $\mathcal{L}^+(i)$*) of an isovalue $i \in \mathbb{R}$ relatively to a PL scalar field $f : \mathcal{M} \to \mathbb{R}$ is the set of points: $\{p \in \mathcal{M} \mid f(p) \leq i\}$ (respectively $\{p \in \mathcal{M} \mid f(p) \geq i\}$).*

Definition 46

(Level set) *The* level-set $f^{-1}(i)$ *of an isovalue $i \in \mathbb{R}$ relatively to a PL scalar field $f : \mathcal{M} \to \mathbb{R}$ is the pre-image of i onto $\mathcal{M}$ through f: $f^{-1}(i) = \{p \in \mathcal{M} \mid f(p) = i\}$.*

Property 7 (Level set) *The level set $f^{-1}(i)$ of a regular isovalue $i \in \mathbb{R}$ relatively to a PL scalar field $f : \mathcal{M} \to \mathbb{R}$ defined on a PL d-manifold $\mathcal{M}$ is a $(d-1)$-manifold.*

Property 8 (Level set) *Let $f : \mathcal{T} \to \mathbb{R}$ be a PL scalar field and σ be a d-simplex of $\mathcal{T}$. For any isovalue $i \in f(\sigma)$, the restriction of $f^{-1}(i)$ within σ belongs to an Euclidean subspace of $\mathbb{R}^d$ of dimension $(d-1)$.*

This latter property directly follows from the property 6 on the gradient of PL scalar fields, which is piecewise constant (a level set is ev-

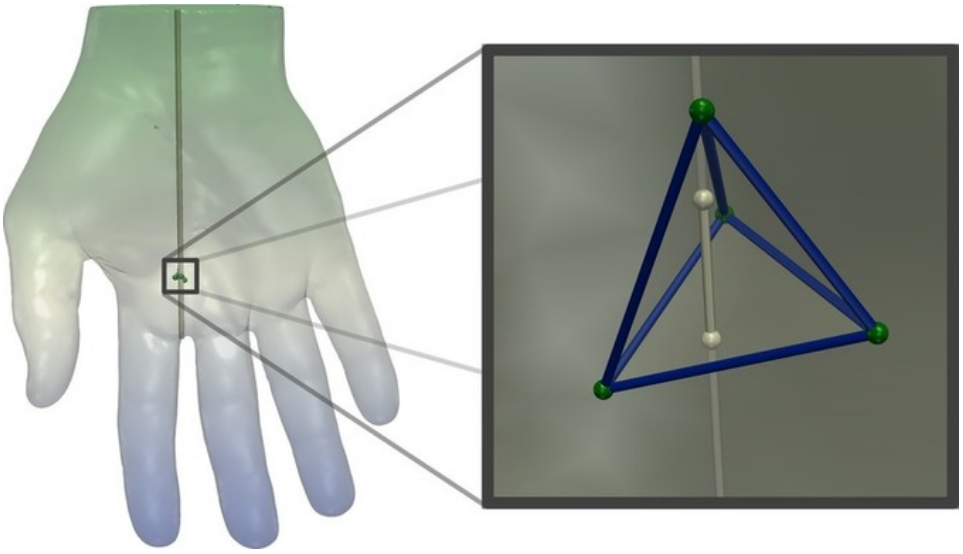


Figure 3.11 – *Examples of integral line (left) of a PL scalar field defined on a PL* 3*-manifold. Right: restriction of the integral line to a* 3*-simplex.*

erywhere orthogonal to the gradient). It follows that the level sets of PL scalar fields defined on PL manifolds can be encoded as PL manifolds, as illustrated with the white PL 2-manifold in Figure 3.10 (right).

Property 9 (Level set) *Let $f : \mathcal{T} \to \mathbb{R}$ be a PL scalar field and σ be a d-simplex of $\mathcal{T}$. For any two isovalues $i \neq j$ belonging to $f(\sigma)$, the restrictions of $f^{-1}(i)$ and of $f^{-1}(j)$ within σ are parallel.*

This property also follows from the property 6 on the gradient of PL scalar fields and is illustrated in Figure 3.10 (right, dark gray isosurfaces), which shows an *isosurface* restricted to a 3-simplex (i.e. a level set of a PL scalar field defined on a PL 3-manifold). Such strong properties (planarity and parallelism) enable to derive robust and easy-to-implement algorithms for level set extraction (called *"Marching Tetrahedra"* for PL 3-manifolds, and *"Marching Triangles"* for PL 2-manifolds).

Definition 47 (Contour) *Let $f^{-1}(i)$ be the level set of an isovalue i relatively to a PL scalar field $f : \mathcal{T} \to \mathbb{R}$. Each connected component of $f^{-1}(i)$ is called a contour.*

Definition 48

(Integral line) *Let $f : \mathcal{M} \to \mathbb{R}$ be a PL scalar field defined on a PL manifold $\mathcal{M}$. An* integral line *is a path $p : \mathbb{R} \to \mathcal{C} \subset \mathcal{M}$ such that $\frac{\partial}{\partial t} p(t) = \nabla f(p(t))$. $lim_{t \to -\infty} p(t)$ and $lim_{t \to \infty} p(t)$ are called the* origin *and the* destination *of the integral line respectively.*

In other words, an integral line is a path which is everywhere tangential to the gradient. In virtue of property 6 on the gradient of PL scalar

fields, it follows that integral lines can be encoded as PL 1-manifolds, as illustrated with the white PL 1-manifold in Figure 3.11 (right).

3.2 Topological abstractions

Level sets (and especially contours) and integral lines are fundamental geometrical objects in Scientific Visualization for the segmentation of regions of interests (burning flames in combustion, interaction pockets in chemistry, etc.) or the extraction of filament structures (galaxy backbones in cosmology, covalent interactions in chemistry, etc.).

Intuitively, the key idea behind Topological Data Analysis is to segment the data into regions where these geometrical objects are *homogeneous* from a topological perspective, and to summarize these homogeneity relationships into a *topological abstraction*. Such a segmentation strategy enables to access these features more efficiently and to classify them according to application dependent metrics for further processing.

In the following, I introduce such topological abstractions for feature extraction, segmentation and classification purposes.

3.2.1 Critical points

In the smooth setting, critical points are points of a manifold where the gradient of a smooth scalar field vanishes. Unfortunately, this notion does not directly translate into the PL setting since the gradient of a PL scalar field is piecewise constant. This requires to use an alternate definition, which interestingly involves topological and combinatorial reasonings.

Morse theory (Mil63) relates the study of the topology of manifolds to the study of a specific group of smooth scalar fields defined on them (called *Morse* functions). One of the key results of Morse theory is the following property: the Betti numbers of the sub-level sets of a Morse function only change in the vicinity of a critical point. In other words, the topology of the sub-level sets only evolves when crossing a critical point. This observation is at the basis of a formalization of critical points in the PL setting.

Definition 49

(Critical point) *Let $f : \mathcal{M} \rightarrow \mathbb{R}$ be a PL scalar field defined on a PL manifold $\mathcal{M}$. A vertex v of $\mathcal{M}$ is a* regular point *if and only if both $Lk^-(v)$ and $Lk^+(v)$ are simply connected. Otherwise, v is a* critical point *of f and $f(v)$ is called a* critical isovalue *(as opposed to* regular isovalues*).*

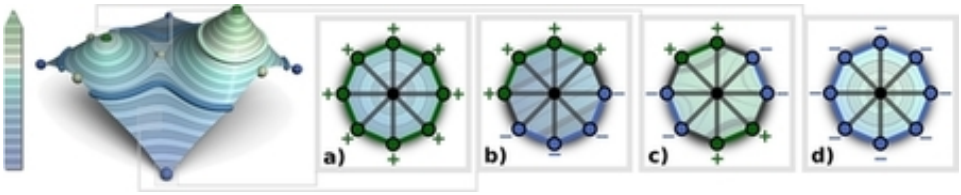


Figure 3.12 – *Scalar field on a terrain (left). A level set is shown in blue, a contour is shown in white. Vertices can be classified according to the the connectivity of their lower (blue) and upper (green) links. From left to right: a minimum (a, blue spheres on the left), a regular point (b), a simple saddle (c, white spheres on the left) and a maximum (d, green spheres on the left).*

Definition 50 (Critical contour) *Let $f : \mathcal{M} \to \mathbb{R}$ be a PL scalar field defined on a PL manifold $\mathcal{M}$. A contour of f which contains one of its critical points is called a* critical contour.

In virtue of these definitions and the property 6 on their gradient, PL scalar fields have many nice properties regarding their critical points.

Property 10 (Critical points) *Let $f : \mathcal{M} \to \mathbb{R}$ be a PL scalar field defined on a compact PL manifold $\mathcal{M}$. The set of critical points of f, noted C_f, contains only isolated critical points and its cardinality $|C_f|$ is finite.*

These properties follow from the fact that the gradient of a PL scalar field is piecewise constant: $\hat{f}$ is assumed to be injective, thus any d-simplex with $d \neq 0$ is mapped to a non null gradient vector. Therefore, critical points are isolated and can only occur on vertices. This makes their number finite for compact PL manifolds. This property is essential for a combinatorial reasoning on critical points. Also, note that the above definitions are independent of the dimension of $\mathcal{M}$.

Definition 51 (Extremum) *Let $f : \mathcal{M} \to \mathbb{R}$ be a PL scalar field defined on a PL manifold $\mathcal{M}$. A critical point v is a* minimum *(respectively a* maximum*) of f if and only if $Lk^-(v)$ (respectively $Lk^+(v)$) is empty.*

Definition 52 (Saddle) *Let $f : \mathcal{M} \to \mathbb{R}$ be a PL scalar field defined on a PL manifold $\mathcal{M}$. A critical point v is a* saddle *if and only if it is neither a minimum nor a maximum of f.*

Figure 3.12 illustrates the notion of critical points on a toy example. The evolution in the topology of the level sets can be observed in the right insets, which illustrate vertex stars (the smallest combinatorial neighborhood of a vertex in a triangulation). For a minimum (respectively a maximum) $\beta_0(f^{-1}(i))$ increases (respectively decreases) by one in the vicinity

of the extremum. For a regular point, this number does not evolve when crossing a regular point. When crossing a saddle, the number of connected components of the restriction of $f^{-1}(i)$ to the star of the vertex first decreases by one exactly at the saddle and then increases by one right above it.

Critical points are usually classified according to their index. For PL scalar fields defined on PL 2-manifolds, minima have index 0, saddles index 1 and maxima index 2. As the dimension of the domain increases, the number of types of critical points also increases. For PL scalar fields defined on PL 3-manifolds, minima have index 0, 1-saddles (saddles that locally merge level sets) have index 1, 2-saddles (saddles that locally split level sets) have index 2 and maxima index 3. In the following, we will note C_f^i the set of critical points of f of index i.

Definition 53 (Saddle multiplicity) *Let $f : \mathcal{M} \to \mathbb{R}$ be a PL scalar field defined on a PL manifold $\mathcal{M}$ and let v be a saddle of f. Let k be the maximum value between $\beta_0(Lk^-(v))$ and $\beta_0(Lk^+(v))$. The* multiplicity *of a saddle is equal to $(k-1)$. A saddle of multiplicity 1 is called a* simple saddle. *It is called a* multi-saddle *otherwise, or $(k-1)$-fold saddle, or alternatively a* degenerate critical point.

Definition 54

(PL Morse scalar field) *Let $f : \mathcal{M} \to \mathbb{R}$ be a PL scalar field defined on a PL manifold $\mathcal{M}$. f is a* PL Morse scalar field *if and only if* (i) *all its critical points have distinct f values and* (ii) *f has no degenerate critical point.*

In practice, any PL scalar field can be easily perturbed into a PL Morse scalar field. The first condition can be easily satisfied by forcing $\hat{f}$ to be injective as described in the previous subsection. The second condition can be satisfied by a process called multi-saddle unfolding (EH09), that locally re-triangulates the star of a $(k-1)$-fold saddle into $(k-1)$ simple saddles.

Interestingly, PL Morse scalar fields inherit from most of the properties of their smooth counter-parts. In particular, the Morse-Euler relation, as first shown by Banchoff (Ban70), still holds for PL Morse scalar fields.

Property 11

(Morse-Euler relation) *Let $f : \mathcal{M} \to \mathbb{R}$ be a PL Morse scalar field defined on a closed PL manifold $\mathcal{M}$ of dimension d. Then the* Morse-Euler relation *holds:* $\chi(\mathcal{M}) = \sum_{i=0}^{i=d} (-1)^i |C_f^i|$

This property nicely summarizes the relation between the critical

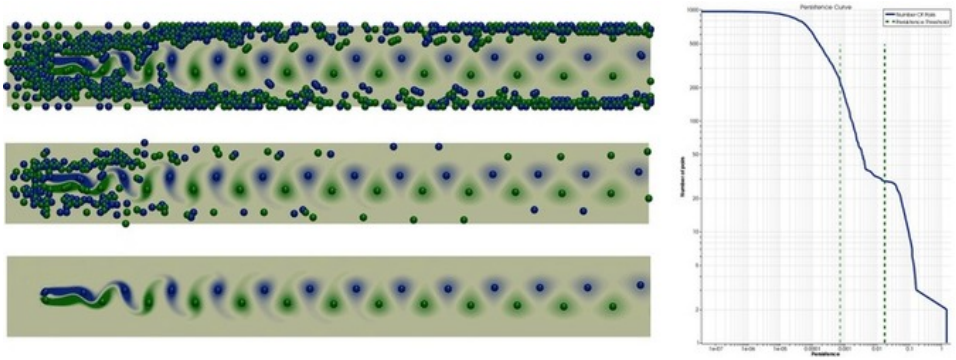


Figure 3.13 – *Minima (blue) and maxima (green) of the orthogonal curl component of a flow simulation of the von Kármán street (flow turbulence behind an obstacle, here at the left of the domain). Right: persistence curve of the field. Selecting extrema involved in pairs more persistent than an increasing threshold (vertical lines, right) yields a hierarchy of critical point sets (left). Here, the light green vertical line (right) corresponds to the middle level (left) while the dark green line (right) corresponds to the bottom level (left). In practice, a flat plateau in the persistence curve (right) often indicates a separation between noise and features.*

points of a PL Morse scalar field and the topology of its domain. Moreover, it also illustrates the global consistency of the local critical point classification definition (definition 49).

In practice, critical points often directly translate into points of interest application wise. For instance, in 2D vector fields obtained in computational fluid dynamics, extrema of the curl of the field indicate the locations of vortices, a high-level notion that has important implications in the efficiency evaluation of a flow (Figure 3.13, bottom).

3.2.2 Notions of persistent homology

As described in the previous subsection, critical points of PL scalar fields can be extracted with a robust, localized yet globally consistent, combinatorial and inexpensive classification (definition 49). However, in practice, this classification strategy will identify, among others, critical points corresponding to slight function undulations coming from the noise in the data generation process (acquisition noise, numerical noise in simulations), as illustrated in Figure 3.13 (top). Therefore, to make critical point extraction reliable and useful in practice, one needs to derive a mechanism to further classify critical points into noise or signal, given some application dependent metric. This is the purpose of Persistent Homology.

Definition 55 (Filtration) *Let $f : \mathcal{K} \to \mathbb{R}$ be an injective scalar field defined on a simplicial complex $\mathcal{K}$ such that $f(\tau) < f(\sigma)$ for each face τ of each simplex σ. Let n be*

Figure 3.14 – *Sub-complexes induced by the filtration of a PL scalar field defined on a PL 3-manifold (dark blue: $\mathcal{L}^-(i)$, light blue: $\mathcal{L}^-(j)$). From left to right: $\beta_0(\mathcal{L}^-(i)) = 3$, $\beta_0(\mathcal{L}^-(j)) = 4$, $\beta_0(\mathcal{L}^-(i,j)) = 2$.*

the number of simplices of $\mathcal{K}$ and let $\mathcal{L}^-(i)$ be the sub-level set of f by the i^{th} value in the sorted set of simplex values. The nested sequence of subcomplexes $\mathcal{L}^-(0) \subset \mathcal{L}^-(1) \subset \cdots \subset \mathcal{L}^-(n-1) = \mathcal{K}$ is called the filtration *of f.*

The general notion of filtration is preferred here to the more specific notion of lower star filtration (specifically adapted to PL scalar fields) as this general introduction will be useful in the next subsections.

Definition 56 (Homomorphism) *A* homomorphism *is a map between groups that commutes with the group operation.*

For instance, the group operation for the group of p-chains is the formal sum of p-simplices (see definition 34).

The filtration of a scalar field f induces a sequence of homomorphisms between the homology groups of the subcomplexes of $\mathcal{K}$:

$$H_p(\mathcal{L}^-(0)) \to H_p(\mathcal{L}^-(1)) \to \cdots \to H_p(\mathcal{L}^-(n-1)) = H_p(\mathcal{K}) \qquad (3.1)$$

Definition 57 (Persistent homology group) *The p^{th}* persistent homology groups *are the images of the homomorphisms induced by inclusion, noted $H_p^{i,j}$, for $0 \leq i \leq j \leq n-1$. The corresponding p^{th}* persistent Betti numbers *are the ranks of these groups, $\beta_p^{i,j} = rank(H_p^{i,j})$.*

Figure 3.14 provides a visual interpretation of the notion of 0^{th} persistent Betti number, which characterizes connected components. Given two nested sub-complexes $\mathcal{L}^-(i)$ and $\mathcal{L}^-(j)$, with $i < j$, the sub-complex induced by inclusion with regard to the 0^{th} homology group is noted $\mathcal{L}^-(i,j)$: it is defined by the connected components of $\mathcal{L}^-(j)$ which have

non empty intersections with these of $\mathcal{L}^-(i)$ (which *includes* them). The 0^{th} homology group of $\mathcal{L}^-(i,j)$ is composed of the classes of the 0^{th} homology group that already existed at the i^{th} isovalue and which still exist at the j^{th} isovalue.

In this example, only 2 of the 4 connected components of $\mathcal{L}^-(j)$ include connected components of $\mathcal{L}^-(i)$. Therefore, among the 3 connected components of $\mathcal{L}^-(i)$, only 2 of them are *persistent* in the interval $[i,j]$.

Therefore, persistent homology provides a mechanism to characterize the importance of topological features (here connected components) with regard to a specific measure (here a PL scalar field), at multiple scales (here the interval $[i,j]$). This interpretation can be generalized to other topological features (for PL 3-manifolds cycles and voids) by extending it to other Betti numbers.

The previous example illustrated the case where a class of the 0^{th} homology group (representing a connected component) disappeared in between the i^{th} and j^{th} isovalues: $\beta_0(\mathcal{L}^-(i)) = 3$, $\beta_0(\mathcal{L}^-(i,j)) = 2$. One can further track the precise isovalue where classes appear, disappear or merge with others. Such events correspond to a change in the Betti numbers of the sub-level set of the scalar field. As discussed in the previous subsection, these changes occur at critical points. Therefore, the notions of *birth* and *death* of classes of persistent homology groups can be associated with pairs of critical points of the input scalar field, and the absolute value of their f value difference is called the *persistence* of the pair.

In particular, the merge of two classes represent a *death* event. In such a case, the least persistent class (the youngest of the two) is chosen as the dying class. This choice is often called the *Elder's rule* (EH09). Once this is established, one can pair without ambiguity all the critical points of a scalar field. Note that this observation could already be foreseen with the Morse-Euler relation. For instance, the removal of an index 2 critical point (a maximum) of a PL Morse scalar field defined a PL 2-manifold implies the removal of a paired index 1 critical point (a saddle) in order to keep the Euler characteristic constant (property 11).

Therefore, it is possible to enumerate all the classes of all p^{th} homology group by enumerating the critical point pairs identified with the above strategy. This list of critical point pairs can be concisely encoded with a topological abstraction called the *Persistence Diagram* (Figure 3.15), noted $\mathcal{D}(f)$. This diagram is a one-dimensional simplicial complex that embeds each pair in $\mathbb{R}^2$ by using its *birth* value as a first component and its *birth* and *death* values as second components. The persistence diagram comes

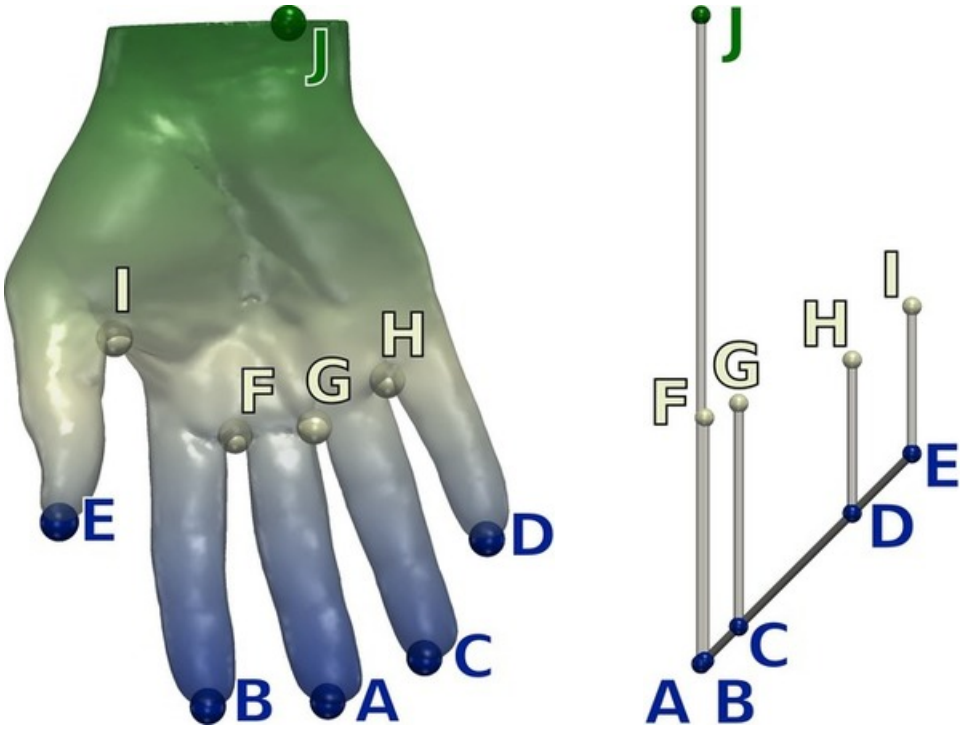


Figure 3.15 – *Critical points of a PL scalar field f defined on a PL 3-manifold (left) and its persistence diagram $\mathcal{D}(f)$ (right). In the diagram, each pair of critical points is represented by a white bar and its persistence is given by the height of the bar.*

with several interesting properties. In particular, the stability theorem (CSEH05) states that given two PL scalar fields f and g defined on a common domain, the bottleneck distance between their persistence diagrams is bounded by the difference between the two functions with regard to the infinity norm: $d_B(\mathcal{D}(f), \mathcal{D}(g)) \leq ||f - g||_\infty$. Intuitively, this means that given a slight perturbation of a scalar field, its persistence diagram will only slightly vary. This stability result further motivates the usage of the persistence diagram as a stable topological abstraction of a scalar field (used for instance in function comparison).

In practice however, an alternate representation is often preferred to isolate critical point pairs corresponding to important features from these corresponding to noise. The *Persistence Curve*, noted $\mathcal{C}(f)$, is a diagram that plots the number of critical point pairs with persistence higher than a threshold ϵ, as a function of this threshold ϵ. When displayed in logarithmic scale, such curves often exhibit a flat plateau separating features with very low persistence from these of higher persistence (see Figure 3.13). In practice, such a plateau is instrumental to manually identify a relevant persistence threshold for the user-driven selection of the most important critical points of the field, as illustrated in Figure 3.13 (bottom). Also, note

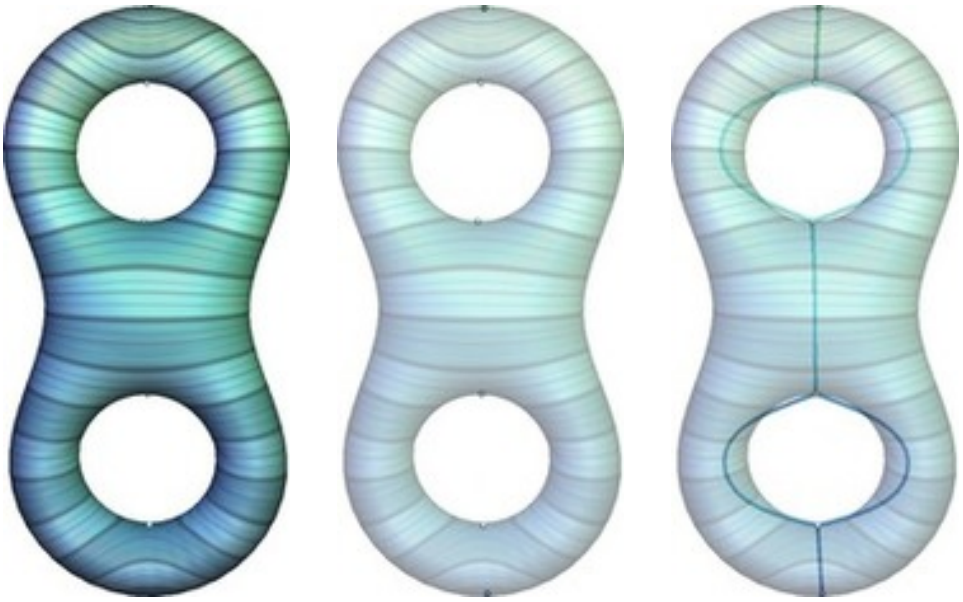


Figure 3.16 – *PL Morse scalar field defined on a PL 2-manifold (left and center) and its Reeb graph (right).*

that extracting the critical point pairs more persistent than a threshold ϵ for increasing values of ϵ yields a hierarchy of sets of critical points, that enables to interactively explore them at multiple scales of importance, as showcased throughout Figure 3.13.

3.2.3 Reeb graph

The persistence curve and the persistence diagrams provide concise representations of the critical point pairs of a PL scalar field, along with their persistence. However, they do not provide any information related to the adjacency relations of these pairs on the domain. This is the purpose of more advanced topological abstractions, such as the Reeb graph (Ree46).

Definition 58

(Reeb graph) *Let $f : \mathcal{M} \to \mathbb{R}$ be a PL Morse scalar field defined on a compact PL manifold $\mathcal{M}$. Let $f^{-1}(f(p))_p$ be the contour of f containing the point $p \in \mathcal{M}$. The Reeb graph $\mathcal{R}(f)$ is a one-dimensional simplicial complex defined as the quotient space on $\mathcal{M} \times \mathbb{R}$ by the equivalence relation $(p_1, f(p_1)) \sim (p_2, f(p_2))$, which holds if:*

$$\begin{cases} f(p_1) = f(p_2) \\ p_2 \in \left(f^{-1}(f(p_1))\right)_{p_1} \end{cases}$$

The Reeb graph can also be defined alternatively as the *contour retract* of $\mathcal{M}$ under f (a continuous map that retracts each contour to a single

point, such that its image is a subset of its domain and its restriction to its image is the identity). Note that f can be decomposed into $f = \psi \circ \phi$, where $\phi : \mathcal{M} \to \mathcal{R}(f)$ is the contour retraction and $\psi : \mathcal{R}(f) \to \mathbb{R}$ is a continuous function that maps points of $\mathcal{R}(f)$ to their f value in $\mathbb{R}$.

Intuitively, as suggested by the previous definition, the Reeb graph continuously contracts each connected component of level sets to a point, yielding a one-dimensional simplicial complex that can be optionally embedded in $\mathbb{R}^3$, as illustrated in Figure 3.16.

Since the Betti numbers of the level sets change at critical points (in particular β_0 may change), the Reeb graph has a tight connection with the critical points of the function. In particular, branching occurs when $\beta_0(f^{-1}(i))$ changes as i evolves, as further detailed below:

Property 12 (Images through ϕ (Ree46)) *Let $\mathcal{R}(f)$ be the Reeb graph of a PL Morse scalar field $f = \psi \circ \phi$ defined on a PL d-manifold. Let the valence of a 0-simplex $v \in \mathcal{R}(f)$ be the number of 1-simplices in its star $St(v)$.*

- *All regular points of f map through ϕ to a point in the interior of a 1-simplex of $\mathcal{R}(f)$. The inverse is true.*
- *All critical points of index 0 or d (all extrema of f) map through ϕ to 0-simplices of $\mathcal{R}(f)$ of valence 1. The inverse is true.*
- *If $d = 2$, all critical points of index 1 (all saddles of f) map through ϕ to 0-simplices of $\mathcal{R}(f)$ of valence 2, 3 or 4. The inverse is true.*
- *If $d \geq 3$, all critical points of index 1 or $(d-1)$ (subsets of saddles of f) map through ϕ to 0-simplices of $\mathcal{R}(f)$ of valence 2 or 3. The inverse is not necessarily true.*
- *If $d > 3$, all critical points of index different from 0, 1, $(d-1)$ or d map through ϕ to 0-simplices of $\mathcal{R}(f)$ of valence 2. The inverse is not necessarily true.*

The original description of the Reeb graph (Ree46) (including the above properties) implies that all critical points of f map to 0-simplices of $\mathcal{R}(f)$. In more contemporary descriptions, two 1-simplices sharing a valence-2 0-simplex as a face are considered to form only one 1-simplex. Therefore, in the contemporary vision of the Reeb graph, extrema map to valence-1 vertices while only the saddles where $\beta_0(f^{-1}(i))$ evolves map to vertices of higher valence. In particular, saddles where $\beta_0(f^{-1}(i))$ decreases (respectively increases) are called *join* (respectively *split*) saddles. For PL

3-manifolds, join (respectively split) saddles have index 1 (respectively 2). In this vision, not all 1 and 2-saddles map to vertices of the Reeb graph.

Since the Reeb graph has a tight connection with the critical points of its scalar field, some of the properties of PL Morse scalar fields translate in the Reeb graph setting.

Definition 59 (Loops in a Reeb graph) *Let $\mathcal{R}(f)$ be the Reeb graph of a PL Morse scalar field f defined on a PL d-manifold $\mathcal{M}$. Each independent cycle of $\mathcal{R}(f)$ is called a* loop. *The number of loops of a Reeb graph is noted $l(\mathcal{R}(f))$.*

The two saddles of each loop with the highest and lowest ψ values are usually called *loop saddles*.

Property 13 (Loops in a Reeb graph) *Let $\mathcal{R}(f)$ be the Reeb graph of a PL Morse scalar field f defined on a compact PL d-manifold $\mathcal{M}$. $l(\mathcal{R}(f))$ is bounded by $\beta_1(\mathcal{M})$:*

$$l(\mathcal{R}(f)) \leq \beta_1(\mathcal{M})$$

As discussed by Cole-McLaughlin et al. (CMEH*03), this property follows from the fact that the construction of the Reeb graph can lead to the removal of 1-cycles of $\mathcal{M}$, but not to the creation of new ones. In the case of PL 2-manifolds, tighter bounds have be shown (CMEH*03).

Property 14 (Loops in a Reeb graph on PL 2-manifolds) *Let $\mathcal{R}(f)$ be the Reeb graph of a PL Morse scalar field f defined on a PL 2-manifold $\mathcal{M}$. Let $b(\mathcal{M})$ be the number of connected components of $\mathcal{M}$ and $g(\mathcal{M})$ its genus. The number of loops of $\mathcal{R}(f)$ can be described as follows:*

- *If $\mathcal{M}$ is orientable (admits a non-null and continuous normal vector field):*
  - *if $b(\mathcal{M}) = 0$, then $l(\mathcal{R}(f)) = g(\mathcal{M})$;*
  - *otherwise $g(\mathcal{M}) \leq l(\mathcal{R}(f)) \leq 2g(\mathcal{M}) + b(\mathcal{M}) - 1$*
- *otherwise,*
  - *if $b(\mathcal{M}) = 0$, then $0 \leq l(\mathcal{R}(f)) \leq g(\mathcal{M})/2$*
  - *otherwise $0 \leq l(\mathcal{R}(f)) \leq g(\mathcal{M}) + b(\mathcal{M}) - 1$*

Figure 3.16 illustrates this property for a closed and orientable 2-manifold of genus 2 (here $\chi(\mathcal{M}) = 2 - 2g(\mathcal{M}) - b(\mathcal{M})$) and a Reeb graph having consequently two loops.

It follows from property 13 that the Reeb graph of a PL Morse scalar field defined on a simply-connected PL d-manifold $\mathcal{M}$ is loop free ($\beta_1(\mathcal{M}) = 0$). In this specific case, the Reeb graph is usually called a *Contour tree* and is noted $\mathcal{T}(f)$. Variants of the Reeb graph, called the *Join*

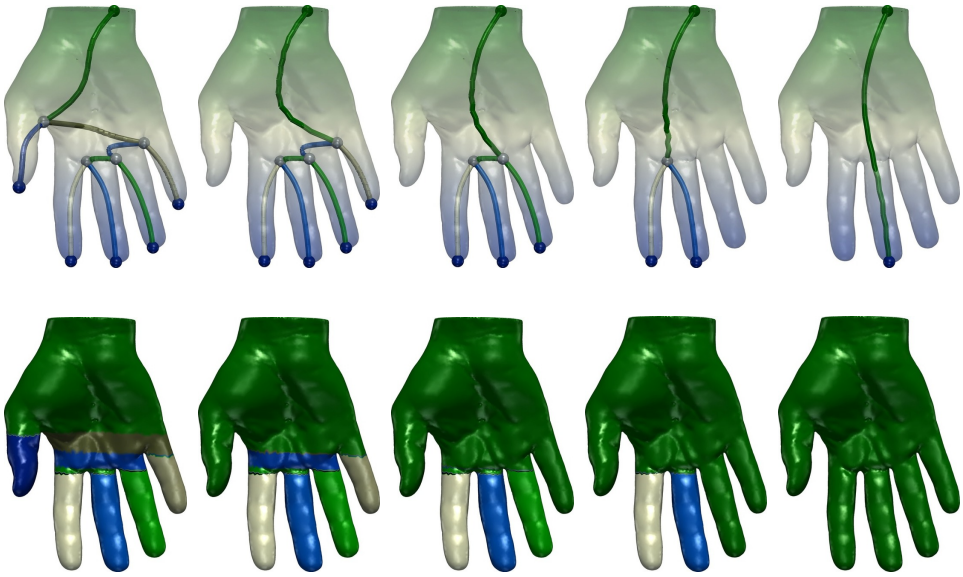


Figure 3.17 – *Hierarchy of Reeb graphs obtained by repeated persistence-driven removal of their* 1*-simplices (top) and hierarchy of data segmentations (bottom) obtained by considering the pre-image by* ϕ *of each* 1*-simplex of the Reeb graphs (matching colors).*

(respectively *Split*) *trees* are defined similarly by contracting connected components of sub (respectively sur) level sets to points (instead of level sets) and are noted $\mathcal{J}(f)$ (respectively $\mathcal{S}(f)$). Note that the join (respectively split) tree of a PL Morse scalar field admitting only one maximum (respectively minimum) is equal to its contour tree.

Since the Reeb graph is a one-dimensional simplicial complex, a filtration of $\psi : \mathcal{R}(f) \to \mathbb{R}$ can be considered and therefore persistent homology concepts (previous subsection) readily apply to the Reeb graph without specialization. In particular, one can directly read the $(0, 1)$ critical point pairs of f (minima and join saddles) from the join tree by removing its 1-simplices attached to a minimum, one by one in order of their persistence, as illustrated in Figure 3.17 (top). A similar strategy applied to the split tree enumerates all $(d-1, d)$ critical point pairs ($(d-1)$-saddles and maxima). The time-efficient enumeration of these types of critical point pairs is one of the primary applications of the Reeb graph in practice. Note that simplifying in such a way the Reeb graph, similarly to the critical points in the previous subsection, yields a hierarchy of Reeb graphs (for each of which a corresponding, simplified, PL Morse scalar field is guaranteed to exist), that enables to interactively explore it at multiple scales of importance, as showcased throughout Figure 3.17 (top).

Finally, note that the pre-image by ϕ of $\mathcal{R}(f)$ induces a complete partition of $\mathcal{M}$. In particular, the pre-image of a 1-simplex $\sigma \in \mathcal{R}(f)$ is guaranteed by construction to be connected (ϕ^{-1} is implemented in practice by

marking during the construction of $\mathcal{R}(f)$ each vertex with the identifier of the 1-simplex where it maps to). This latter property is instrumental in various tasks in scientific visualization, including the efficient indexing of contours (for fast level set extraction) or the automatic and interactive data segmentation into regions of interest (especially when feature boundaries coincide with level sets). This latter capability of the Reeb graph can be nicely combined with persistent homology concepts, yielding hierarchies of data segmentations, as illustrated in Figure 3.17 (bottom).

3.2.4 Morse-Smale complex

As discussed in the previous subsection, in the modern interpretation of the Reeb graph, not all critical points are captured as 0-simplices. Therefore, the Reeb graph only describes the adjacency relations of a sub-set of critical point pairs. To capture such exhaustive adjacency relations, one needs to consider another topological abstractions, called the Morse-Smale complex, that is constructed by considering equivalence classes on integral lines instead of contours (see (Gyu08) for further details).

Property 15 (Integral lines) *Let $f : \mathcal{M} \to \mathbb{R}$ be a PL Morse scalar field defined on a closed PL d-manifold $\mathcal{M}$. Then, the following properties hold:*

- *Two integral lines are either disjoint or the same;*
- *Integral lines cover all of $\mathcal{M}$;*
- *The origin and the destination of an integral line are critical points of f.*

The latter property is particularly interesting. It means that an integral line can be characterized by its extremities, which are guaranteed to be critical points of f. Then, one can introduce an equivalence relation that holds if two integral lines share the same extremities. This is the key idea behind the Morse-Smale complex.

Definition 60 (Ascending manifold) *Let $f : \mathcal{M} \to \mathbb{R}$ be a PL Morse scalar field defined on a PL d-manifold $\mathcal{M}$. The* ascending *(respectively* descending*)* manifold *of a critical point p of f is the set of points belonging to integral lines whose origin (respectively destination) is p.*

Property 16 (Ascending manifolds) *Let $f : \mathcal{M} \to \mathbb{R}$ be a PL Morse scalar field defined on a PL d-manifold $\mathcal{M}$. Let p be an index-i critical point of f. The ascending (respectively descending) manifold of p is an open set of $\mathcal{M}$ of dimension $(d - i)$ (respectively i).*

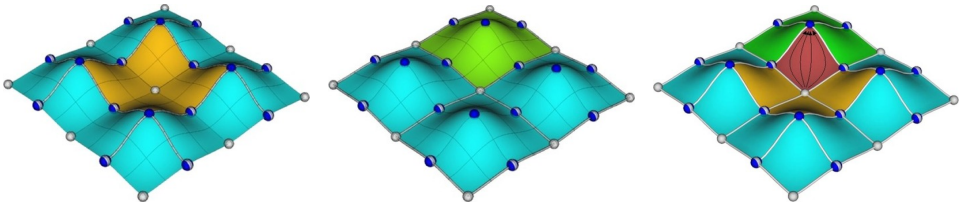


Figure 3.18 – *Ascending (left) and descending (center) manifolds and Morse-Smale complex (right) of a PL Morse scalar field f defined on a PL 2-manifold. Image taken from (Gyu08).*

Figure 3.18 illustrates these properties with a PL Morse scalar field defined on a PL 2-manifold $\mathcal{M}$. In particular, the ascending manifold of a minimum is a subset of $\mathcal{M}$ of dimension 2 (shown in orange, left). Similarly, the descending manifold of a maximum is also a subset of $\mathcal{M}$ of dimension 2 (show in green, center). For PL 2-manifolds, in both cases, ascending and descending manifolds of saddles have dimension 1 (white integral lines in both images).

Definition 61 (Morse complex) *Let $f : \mathcal{M} \to \mathbb{R}$ be a PL Morse scalar field defined on a PL d-manifold $\mathcal{M}$. The complex formed by all descending manifolds of f is called the* Morse complex.

Given this definition, the complex of all ascending manifolds shown on the left of Figure 3.18 is the Morse complex of $-f$, while the complex of all descending manifolds shown in the center is that of f.

Definition 62 (Morse-Smale function) *Let $f : \mathcal{M} \to \mathbb{R}$ be a PL Morse scalar field defined on a PL d-manifold $\mathcal{M}$. f is a* Morse-Smale *function if the ascending and descending manifolds only intersect transversally.*

Intuitively, the transversal intersection condition implies that ascending and descending manifolds are not parallel at their intersection (this condition is enforced in practice through local remeshing). This implies that when these intersect exactly at one point, such a point is critical. This also implies that given an integral line, the index of its origin is smaller than that of its destination.

Definition 63 (Morse-Smale complex) *Let $f : \mathcal{M} \to \mathbb{R}$ be a Morse-Smale scalar field defined on a PL d-manifold $\mathcal{M}$. The complex formed by the intersection of the Morse complex of f and that of $-f$ is called the* Morse-Smale *complex and noted $\mathcal{MS}(f)$.*

Figure 3.18 (right) illustrates such an intersection. As shown with the red region, all integral lines (black curves) of a given cell of the complex (irrespectively of its dimension) share the same origin and destination.

Note that one can derive a simplicial decomposition of the Morse-Smale complex by subdividing each d-dimensional cell into valid d-simplices. By construction, all the critical points of f will therefore map to distinct 0-simplices of such a decomposition, since the Morse-Smale complex captures all ascending and descending manifolds (and therefore all critical points). Then, as for the Reeb graph (previous subsection), persistent homology concepts also readily apply to the simplicial decomposition of the Morse-Smale complex and simplifying it for increasing values of persistence also yields a hierarchy that enables to interactively explore the Morse-Smale complex at multiple scales of importance.

Finally, by construction, the Morse-Smale complex provides a partition of the domain that is instrumental in scientific visualization, especially when features or their boundaries coincide with the gradient. Alike the Reeb graph, this segmentation capability can be nicely combined with persistent homology concepts, yielding hierarchies of data segmentations, as detailed in the following subsection.

3.3 Algorithms and applications

In this section, I briefly discuss the state-of-the art algorithms for computing the topological abstractions described above, and I also briefly introduce some of their applications.

Persistent homology

Critical points of PL scalar fields are usually extracted with a simple, robust, localized yet globally consistent, and easily parallelizable algorithm that directly implements the definitions presented in Section 3.2.1, and which derive from a seminal paper by Banchoff (Ban70).

Critical point pair extraction as well as their persistence evaluation (Section 3.2.2) are usually implemented through sparse matrix reduction (ELZ02) with an algorithm with $O(n^3)$ worst case time complexity (where n is the number of simplices). Note however, that for the purpose of feature selection, only extrema-saddle pairs seem to have a practical interest and these can be computed more efficiently with the Reeb graph as described previously.

Persistence diagrams (which encode all critical point pairs along with their persistence) have been widely used for the purpose of function comparison, especially for high-dimensional domains where more advanced

topological abstractions are more difficult to compute and simplify (see for instance (Ghr07), (CCSG*09) and (RL15)).

Cohen-Steiner et al. (CSEH05) showed that the bottleneck distance between the persistence diagrams of two PL scalar fields f and g computed on a common domain was bounded by the distance between the two functions with regard to the infinity norm ($||f-g||_\infty$). This result raises the reciprocal question: *given a persistence diagram $\mathcal{D}(f)$ where all pairs less persistent than a threshold ϵ have been removed (noted $\mathcal{D}(g)$), can we compute a function g sufficiently close from f that admits $\mathcal{D}(g)$ as persistence diagram?* This question has major practical implications since the time complexity of the algorithms for the construction or processing of topological abstractions is often dictated by the number of critical points in the input scalar field. Often in practice, it is possible to easily discriminate critical points that are not relevant application-wise. Therefore, there exists an applicative interest for an efficient pre-processing of an input scalar field, that would minimally perturb it to remove a given set of critical points. This question has first been addressed in the case of PL scalar fields defined on PL 2-manifolds by Edelsbrunner et al. (EMP06), who showed that such a function g existed and that its difference to the input was bounded by ϵ: $||f-g||_\infty \leq \epsilon$. These authors also provided an algorithm to compute it. However this algorithm is complicated and difficult to implement. Moreover, as persistence pairs are processed in order of their highest extremity, the same vertices are swept several times when canceable persistence pairs are nested. Attali et al. (AGH*09) and Bauer et al. (BLW12) presented independently a similar approach to this problem for filtrations and Discrete Morse functions. However, converting the output of these algorithms to PL scalar fields (which is the standard scalar field representation for many applications) requires an important subdivision of the domain. Also, these approaches only deal with closed surfaces. We introduced in 2012 a general algorithm (TP12) for the topological simplification of PL scalar fields on closed or open PL 2-manifolds, capable of removing arbitrary critical point pairs (not necessarily the least persistent), which enables the usage of application-dependent metrics for feature selection. Thanks to its speed, ease of implementation, robustness and generality, we consider this algorithm as the reference for the problem of topological simplification of scalar data on surfaces.

A survey on the concepts of Persistent homology and their applications can be found in (EH08).

Reeb graph

Reeb graphs have been introduced in Computer Science independently by Boyell and Ruston (BR63) (for simply connected surfaces) and by Shinagawa et al. (SKK91).

Efficient algorithms for their computation have first been investigated in the simpler cases of simply connected domains (for which the Reeb graph is called the contour tree). Time-efficient algorithms have first been introduced for 2D domains (vKvOB*97), then 3D domains (TV98) and last for domains of arbitrary dimension, with an algorithm (CSA00) with optimal time complexity: $O(|\sigma_0|log(|\sigma_0|) + (\sum_{i=0}^{i=1}|\sigma_i|)\alpha(\sum_{i=0}^{i=1}|\sigma_i|))$, where $|\sigma_i|$ is the number of i-simplices in the domain and $\alpha()$ is an extremely slowly growing function (i.e. the inverse of the Ackermann function). In particular, the latter algorithm is considered by the community as the reference for the problem of contour tree computation thanks to its optimal time complexity, its ease of implementation, its robustness, and its practical performances. An open-source implementation is also available (Dil07). This algorithm first computes the join and split trees by tracking the merge events of a union-find data-structure, by processing the vertices in ascending (respectively descending) order. Last, the two trees are combined to form the contour tree in a linear pass.

Regarding more general domains, an algorithm (CMEH*03) has been introduced for PL scalar fields defined on PL 2-manifolds, with optimal time complexity: $O(|\sigma_1|log(|\sigma_1|)$. A more recent, non-optimal algorithm (PSF08) would be recommend instead however, due to its ease of implementation and acceptable performances in practice. This algorithm explicitly constructs the partition induced by the pre-image of ϕ to retrieve the 1-simplices of $\mathcal{R}(f)$, by computing the critical contours of all saddles of f as boundaries. This makes the algorithm output-sensitive but leads to a worst-case complexity of $O(|\sigma_0| \times |\sigma_2|)$.

Regarding higher dimensional (non simply-connected) domains, several attempts have been proposed, especially with the more practical target of PL 3-manifolds in mind. However, many properties of the 2-dimensional domains exploited by the above algorithms do not hold for 3-dimensional domains, making the problem more challenging. Pascucci et al. (PSBM07) introduced the first algorithm for the computation of the Reeb graph on domains of arbitrary dimension. Its streaming nature however, while appealing in specific applications, makes it implementation difficult. An open-source implementation, which I wrote in 2009 based

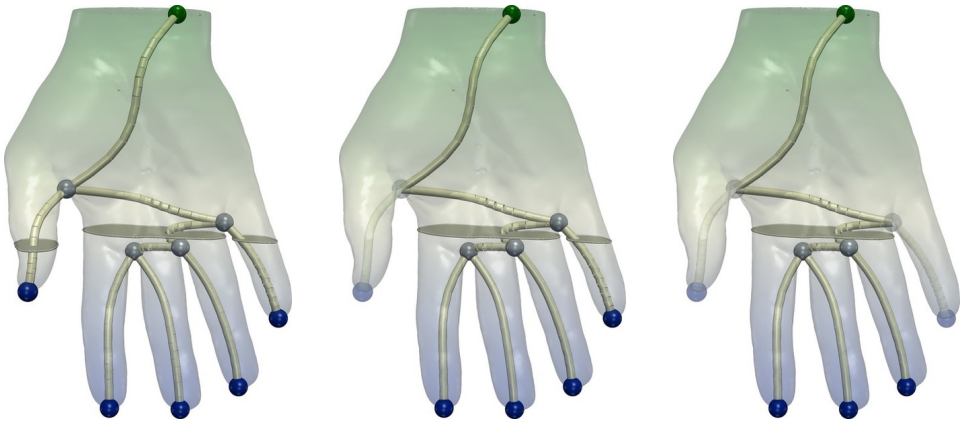


Figure 3.19 – *Topological simplification of isosurfaces (white surface, middle). The* 1*-simplices of* $\mathcal{R}(f)$ *that are less persistent than an increasing threshold (transparent, from left to right) are not considered for contour seed extraction, yielding a progressive removal of the least prominent contours from the isosurface.*

on Giorgio Scorzelli's original implementation, is available in the official release of the open-source library the Visualization ToolKit (Tie09). Doraiswamy and Natarajan (DN08) extended the quadratic complexity algorithm by Patane et al (PSF08) from PL 2-manifolds to PL 3-manifolds. We introduced the first practical algorithm (TGSP09) (further described in Chapter 4) for the efficient computation of the Reeb graph of PL scalar fields defined on PL 3-manifolds in 2009. Thanks to its speed, it enabled in practice to transfer all of the contour tree based interactive applications to more general non-simply connected domains. We considered this algorithm as the reference for the problem of Reeb graph computation on PL 3-manifolds, until a dimension-independent, optimal time complexity ($O((\sum_{i=0}^{i=2}|\sigma_i|)log(\sum_{i=0}^{i=2}|\sigma_i|))$) algorithm was introduced three years later (Par12).

The contour tree and the Reeb graph have been massively applied in scientific visualization, in particular because of the property that the preimage by ϕ of a 1-simplex $\sigma \in \mathcal{R}(f)$ is guaranteed by construction to be connected. This enables for instance to extract an optimal number of vertex seeds for optimal time level set extraction: each vertex seed initiates the construction of a contour by breadth-first search traversal of the domain (limiting the traversal to the exact set of simplices projecting on the queried isovalue). This seed extraction process requires to store each 1-simplex of $\mathcal{R}(f)$ in a balanced interval tree (CLRS09). At query time, given an isovalue i, all 1-simplices of $\mathcal{R}(f)$ projecting on i can be efficiently retrieved in $O(|\sigma_1|log(|\sigma_1|))$ steps, where $|\sigma_1|$ is here the number of 1-simplices in $\mathcal{R}(f)$. Further, given a 1-simplex $\sigma \in \mathcal{R}(f)$ that projects

on i, a seed vertex can be efficiently extracted if the vertices of the domain projecting to σ through ϕ (let $|\sigma'_0|$ be their number) are stored in a balanced search tree in $O(|\sigma'_0|log(|\sigma'_0|))$ steps. This mechanism can be nicely combined with persistent homology concepts to interactively simplify isocontours, as illustrated in Figure 3.19, by only considering the 1-simplices of $\mathcal{R}(f)$ that are more persistent than a threshold ϵ. Variants of this strategy have been presented in the case of the contour tree by van Kreveld et al. (vKvOB*97) and Carr et al. (CSvdP04). In particular, the latter approach introduced, as an alternative to persistence, several geometrical measures enabling to filter the 1-simplices of $\mathcal{T}(f)$ according to more application-relevant metrics, which reveals to be of major importance in practice.

The partitioning capabilities of the Reeb graph have also been instrumental in scientific visualization for data segmentation tasks, especially in cases where the boundaries of regions of interest coincide with level sets. In that context, the Reeb graph enables (with the fast isosurface extraction algorithm presented above) to rapidly extract and distinguish each of these boundaries, at multiple scales of importance when combined with persistent homology mechanisms. Chapter 6 discusses two applications of these capabilities to combustion (BWT*11) and chemistry (GABCG*14). These segmentation capabilities also serve as the basis of more advanced techniques, for the tracking of features over time (SB06), for the design of transfer functions in volume rendering (WDC*07) or the similarity estimation between data features (TN14).

Apart from scientific visualization, the Reeb graph has also been used as a core data-structure in computer graphics, as discussed in (BGSF08, Tie08).

Morse-Smale complex

The computation of Morse-Smale complexes was first investigated in the case of PL 2-manifolds. An initial algorithm was introduced by Edelsbrunner et al. (EHZ03). This algorithm constructs for each saddle of f the integral lines originating and terminating at the saddle, yielding the set of ascending and descending 1-manifolds. Ascending and descending 2-manifolds are then retrieved through breadth-first search, by growing 2-dimensional regions until ascending or descending 1-manifolds are attained. Further, the authors apply persistent homology concepts to remove the least persistent critical point pairs and describe a mechanism

to update the Morse-Smale complex accordingly (by removing ascending and descending 1-manifolds attached to removed critical points, merging the adjacent 2-manifolds and re-routing their 1-manifold boundaries). Since the number of saddles of f can be proportional to the number of vertices in the domain and since the integral lines can intersect an number of triangles which is proportional to that of the domain, the construction algorithm has a worst case time complexity of $O(|\sigma_0| \times |\sigma_2|)$. This algorithm was latter improved and applied for the first time to scientific visualization by Bremer et al. (BEHP03).

The problem of computing the Morse-Smale complex in higher dimensions is far more challenging. First, as the dimension increases, new types of critical points appear (of increasing index), which translates into the apparition of ascending and descending manifolds of higher and higher dimensions. The construction of each of these types of manifolds implies a quadratic term in the runtime complexity. Second, degenerate cases become more challenging to resolve. This is in particular the case of the resolution of the degenerate critical points and the enforcement of the transversal intersection condition. A complicated algorithm has been proposed for PL scalar fields defined on PL 3-manifolds (EHNP03) but seems highly challenging to implement and no implementation has been reported.

Recently, efficient algorithms have been introduced by walking around the degeneracies of the PL setting and considering a competing formalism, Discrete Morse Theory (For01), which comes with many nice combinatorial properties. In this setting for example, critical points occur on simplices of arbitrary dimension and the index of a critical point coincides with the dimension of its simplex. Moreover, degenerate critical points cannot occur by construction. This formalism considers as *Discrete Morse Functions* scalar fields that map each simplex of the domain to a single scalar value, in such a way that for each simplex σ, there exists at most one simplex for which σ is a face with lower function value, and that there exists at most one simplex that is a face of σ with a higher function value. Then a simplex is critical if no such face and no such coface exist. This formalism additionally introduces the notion of *V-path*, used as an analog to PL integral lines, which is a sequence of pairs of simplices of alternating dimensions (d and $(d+1)$) with descending function values. Then, the notion of *discrete gradient* can be introduced as a pairing of simplices that induces V-paths that are all monotonic and loop-free (Figure 3.20). Given a discrete gradient, critical points correspond to simplices that belong to no

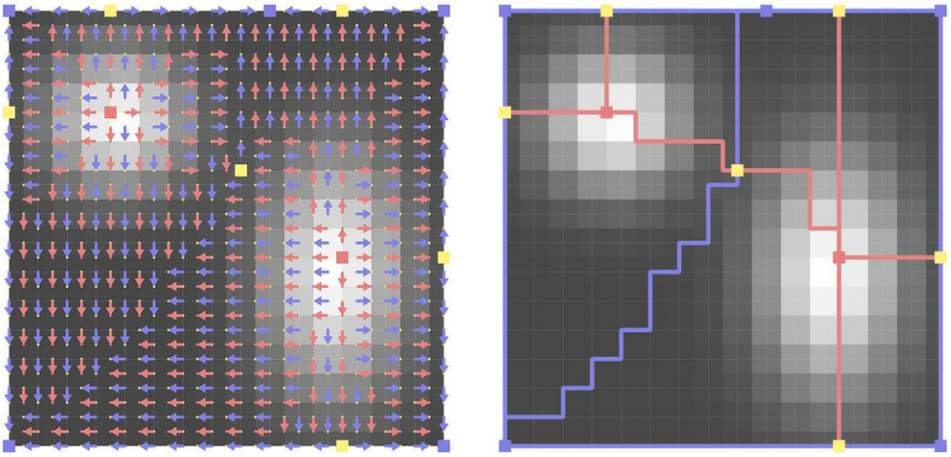


Figure 3.20 – *Discrete gradient field of a simple synthetic function. Blue arrows illustrate the pairing of 0-cells (vertices) with 1-cells (edges) while red arrows show the pairing of 1-cells (edges) with 2-cells (faces). Critical i-cells are shown as blue (d = 0), yellow (d = 1), and red (d = 2) squares. The discrete gradient field defines the MS complex: combinatorial separatrices (blue and red lines) connecting the critical cells.*

simplex pair. As mentioned before, degenerate critical points cannot occur by construction in this setting. Moreover, ascending and descending manifolds are guaranteed to intersect transversally (Gyu08). Therefore, by construction, this formalism avoids all the degeneracies found in the PL setting which make Morse-Smale complex computation challenging. Thus, several efficient algorithms have been proposed for the computation of a discrete gradient from a PL scalar field as well as the construction and simplification of the Morse-Smale complex (GBHP08, RWS11), including a shared-memory parallel algorithm (SN12) whose implementation has been released in open-source (Shi12). Note that this discrete formalism has been described here in the context of PL manifolds for consistency, but it readily applies to arbitrary cellular complexes. This is another factor that motivates its popularity, as regular grids no longer need to undergo simplicial subdivisions in this framework.

The combinatorial consistency of the Discrete Morse Theory setting comes however with a price in terms of geometrical accuracy: in particular, locally, V-paths have to follow simplices of the domain and therefore do not match the integral lines induced by any interpolant. Gyulassy et al. (GBP12) addressed this issue by introducing a novel, probabilistic, discrete gradient construction algorithm whose V-paths are shown to converge to integral lines as the domain sampling increases. We recently improved this approach to avoid the need for domain re-sampling by introducing a discrete gradient construction algorithm that conforms to input

constraints (GGL*14), as further discussed in Chapter 5. When used with integral lines computed through numerical approximation, this gradient construction algorithm yields Morse-Smale complexes whose manifolds better align with the gradient induced by the domain interpolant.

Morse-Smale complexes have been a popular topological abstraction for data analysis and visualization, especially for data segmentation tasks in applications where features of interest (or their boundaries) coincide with the gradient. Then such features can be efficiently captured at multiple scales of importance (thanks to persistent homology mechanisms) by considering the cells of the Morse-Smale complex. This overall strategy has been applied with tailored analysis algorithms to various applications, including the analysis of the Rayleigh-Taylor instability (LBM*06), vortical structures (KRHH11), porous media (GND*07), cosmology (Sou11), combustion (GBG*14), chemistry (GABCG*14), etc. A recent survey (DFFIM15) provides further details regarding the construction, simplification and application of the Morse-Smale complex.

Abstraction

4

Contents

This chapter describes my contributions for the efficient and robust computation of topological abstractions, which play a fundamental role in scientific visualization in order to abstract high-level features from raw scalar data (as described in the previous chapter).

First, I present a combinatorial technique, published in 2012 (TP12), for the topological simplification of scalar data, given some user-defined or application-driven constraints. The algorithm slightly perturbs the input data such that only a constrained sub-set of critical points remains. Thus, this technique can serve in practice as a pre-processing step that significantly speeds up the subsequent computation of topological abstractions.

Second, I present an efficient algorithm, published in 2009 (TGSP09), for the computation of Reeb graphs of PL scalar fields defined on PL 3-manifolds in $\mathbb{R}^3$. This approach described the first practical algorithm for volumetric meshes, with virtually linear scalability in practice and up to 3 orders of magnitude speedups with regard to previous work. Such an algorithm enabled for the first time the generalization of contour-tree based interactive techniques to non simply- connected domains.

The previous chapter introduced the theoretical background required for the reading of this manuscript. In particular, it introduced several topological abstractions that play a fundamental role in scientific visualization for feature extraction and data segmentation.

In this chapter, I described my contributions for the robust and efficient computation of topological abstractions. I first focus on the case of PL scalar fields defined on PL 2-manifolds (with a pre-processing algorithm that speedups subsequent topological abstraction computations), then on the case of PL scalar fields defined on PL 3-manifolds (with a fast Reeb graph computation algorithm).

4.1 Efficient topological simplification of scalar fields

As described in the previous chapter, the computation of the Morse-Smale complex has a quadratic time complexity. This is also the case for popular Reeb graphs algorithms, such as the approach by Patane et al. (PSF08).

However, in practice, it is often easy to discriminate critical points that are non relevant application-wise. Therefore, if one could perturb the input such that such critical points no longer exist, the subsequent computation of topological abstraction would be greatly accelerated. Moreover, while existing schemes for the simplification of topological abstractions produce multi-resolution representations of the abstractions (previous chapter), they do not perform an actual simplification of the underlying scalar field. This can often be useful for further analysis. Finally, in contexts such as scalar field design, it is desirable to obtain a simplified version of the input field directly without having to compute a computationally expensive topological abstraction.

In this section, I present a combinatorial algorithm for the general simplification of scalar fields on surfaces (TP12). This algorithm is simple, fast in practice, and more general than previous techniques. Given a scalar field f, our algorithm generates a simplified function g that provably admits only critical points from a constrained subset of the singularities of f, while guaranteeing a small distance $||f - g||_\infty$ for data-fitting purpose. In contrast to previous combinatorial approaches, our algorithm is oblivious to the strategy used for selecting features of interest and allows critical points to be removed arbitrarily (Fig. 4.1). In the special case where topological persistence is used as a feature identification criteria, our al-

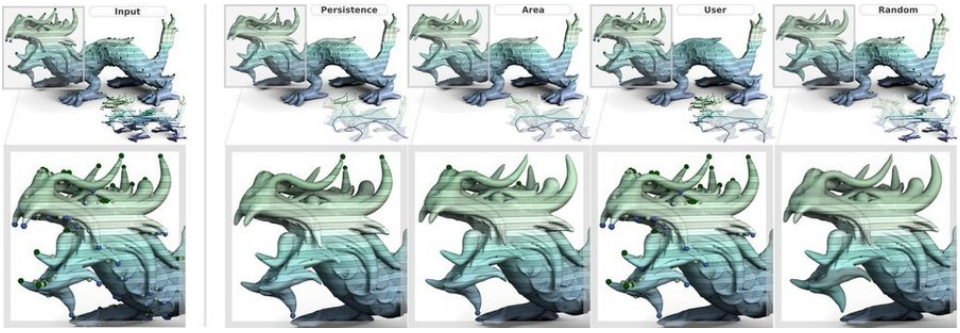


Figure 4.1 – *Given an input scalar field f (left), our combinatorial algorithm generates a simplified function g that provably admits only critical points from a constrained subset of the singularities of f. Our approach is completely oblivious to the employed feature selection strategy, while guaranteeing a small distance $||f - g||_\infty$ for data-fitting purpose. Thus it supports application-dependent simplification scenarios such as the removal of singularities based on local geometrical measures, interactive user selection or even random selection. The topology of the resulting field is summarized with the inset Reeb graphs for illustration purpose.*

gorithm generates a standard ϵ-simplification (EMP06). The algorithm is simple to implement, handles surfaces with or without boundary, and is robust to the presence of multi-saddles (the input is not restricted to true Morse functions). Extensive experiments show the generality of our algorithm as well as its high performance. In particular, the iterative nature of the approach could require a large number of passes but in practice we have not found examples requiring more than five iterations (normally only two are needed). For this reason the experimental results show an $O(n\ log(n))$ practical performance. To demonstrate the use of our approach, we present applications in terrain simplification as well as the acceleration of topological abstraction computation.

This work makes the following contributions:

1. *Approach:* An approach for the topological simplification of scalar fields that does not rely on persistent homology. It yields a simpler, more intuitive, and general setting. We enumerate the critical points that are *non-removable* because of the topology of the domain. We consequently derive a strategy that supports the suppression of arbitrary *removable* critical points of the field. This enables the development of a more general simplification framework than previous approaches, for which ϵ-simplification is a special case.

2. *Algorithm:* An iterative, combinatorial simplification algorithm which is very simple to implement (only a few dozens of lines of C++ code). Given a set of user constraints on the extrema of the

output function, our algorithm automatically identifies and removes the optimal set of saddles with regard to $||f - g||_\infty$, hence guaranteeing a small distance between the input and the output. In contrast to previous approaches, our technique works directly on PL scalar field representations, is robust to multi-saddles, and handles surfaces with or without boundary. The algorithm uses no computationally expensive topological abstraction, such as a Morse-Smale complex or even a contour tree; hence it is very fast in practice. Our extensive experiments on approximated worst-case scenarios show that this iterative algorithm rarely takes more than two iterations to converge.

4.1.1 Preliminaries

In the following, we consider a PL Morse scalar field $f : \mathcal{M} \to \mathbb{R}$ defined on an orientable PL 2-manifold $\mathcal{M}$.

General simplification of scalar fields on surfaces

Definition 64 (General Topological Simplification) *Given a PL scalar field $f : \mathcal{M} \to \mathbb{R}$ with its set of critical points C_f, we call a* general simplification *of f a PL scalar field $g : \mathcal{M} \to \mathbb{R}$ such that the critical points of g form a sub-set of C_f: $C_g \subseteq C_f$ (with identical indices and locations).*

It is often additionally desired that $||f - g||_\infty$ is minimized for data-fitting purpose. In other words, a general simplification consists in constructing a close variant of the input field f from which a set of critical points has been removed. We describe the possible removals:

Closed surfaces The Morse-Euler relation (property 11) defines a dependency between the number of critical points of f and $\chi(\mathcal{M})$, where C_f^i is the set of critical points of f of index i:

$$\chi(\mathcal{M}) = \sum_{i \in \{0,1,2\}} (-1)^i |C_f^i| \; = \; \#_{min(f)} - \#_{saddles(f)} + \#_{max(f)} \tag{4.1}$$

It follows that removing only one extremum, such that the total number of critical points strictly decreases, implies the removal of one saddle (to maintain $\chi(\mathcal{M})$ invariant) and reciprocally. In other words, the removal of the saddles of f are dependent on the removal of its extrema. Certain saddles of f cannot be removed:

$$\chi(\mathcal{M}) = 2 - 2g(\mathcal{M}) = \#_{min(f)} - \#_{saddles(f)} + \#_{max(f)} \tag{4.2}$$

Figure 4.2 – *Non removable critical points: (a) A global minimum and a global maximum have to be maintained for the field not to be constant. (b)* $2\,g(\mathcal{M})$ *saddles cannot be removed. Each boundary component has* 2 *non-removable global* stratified *extrema, which turn into non-removable saddles (c) or (possibly)* exchangeable *extrema (d).*

It follows that f counts exactly $2g(\mathcal{M})$ non-removable saddles, located along the $g(\mathcal{M})$ handles of the surface (Fig. 4.2(b)).

Surfaces with boundary The above properties are valid for surfaces with boundary. In addition, for each boundary component $\mathcal{B} \subseteq \partial\mathcal{M}$, certain saddles cannot be removed. Let $f_\mathcal{B}$ be the restriction of f to $\mathcal{B}$, and $C_{f_\mathcal{B}}$ its critical points that we call *stratified* critical points. By construction, $\mathcal{B}$ is a closed PL 1-manifold. Then:

$$\chi(\mathcal{B}) = \sum_{i\in\{0,1\}} (-1)^i |C^i_{f_\mathcal{B}}| \;=\; \#_{min(f_\mathcal{B})} - \#_{max(f_\mathcal{B})} = 0 \tag{4.3}$$

It follows that $\mathcal{B}$ has an even number of stratified critical points. These cannot be regular points of f on $\mathcal{M}$. For instance, if a maximum of $f_\mathcal{B}$ is only surrounded on the interior of $\mathcal{M}$ by vertices with higher f values (Fig. 4.2(c), right), its lower link is by construction not simply connected; therefore it turns into a saddle of f. If it is only surrounded by vertices with lower f values (Fig. 4.2(d), right), then it turns into a maximum of f (otherwise it is a multi-saddle but f is assumed so far to have only simple saddles). The symmetric property holds for the minima of $f_\mathcal{B}$. Since f is required to admit distinct values for each vertex, $f_\mathcal{B}$ admits a pair of global stratified extrema (Fig. 4.2(c), middle). Consequently, each boundary component of $\mathcal{M}$ necessarily has a pair of critical points of f. If these two points are extrema, they can be removed only if they are not the only extrema of f, leaving a necessary saddle in place in exchange. Otherwise, these necessary critical points are non-removable saddles of f (Fig. 4.2(c), left). In conclusion, the removal of the saddles of f is completely dependent on the removal of its extrema, and for particular cases (summarized in Fig. 4.2) certain critical points are non-removable.

Surface scalar field constrained topology

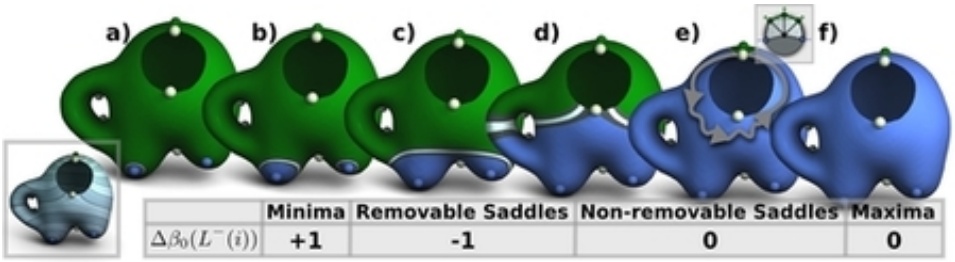
As discussed in the previous sub-section, the removal of the saddles of f is dependent on the removal of its extrema. Then, given the sets of input constraints C_g^0 and C_g^2 (the extrema of g), the target of general simplification is to constrain the topology of the level lines of f such that the output field g is close to f and admits as saddles a valid subset of C_f^1 (such that the Morse-Euler relation still holds, Eq. 4.1). To constrain the topology of the level lines $f^{-1}(i)$ of f, our strategy is to constrain the topology of both the sub- and sur-level sets of f ($\mathcal{L}^-(i)$ and $\mathcal{L}^+(i)$ respectively), which is more practical to achieve. We show in the following that, for surfaces, controlling the connectivity only of $\mathcal{L}^-(i)$ and $\mathcal{L}^+(i)$ is sufficient to enforce the removal (or the preservation) of the removable critical points of f.

Let $f^- : \mathcal{M} \to \mathbb{R}$ be a PL Morse scalar field with only one maximum and several minima (Fig. 4.3). Since f^- has only one maximum, $\beta_0(\mathcal{L}^+(i)) = 1$ for all the i values under the maximum (each vertex of $\mathcal{L}^+(i)$ has a non-empty upper link and thus admits a connected path to the maximum).

Minima A minimum at isovalue i has an empty lower link; then there exists no connected path on $\mathcal{L}^-(i)$ linking this minimum to other lower minima. Thus, as i changes continuously in $\mathbb{R}$, when passing through a minimum of f^-, a new connected component of $\mathcal{L}^-(i)$ has to be created and $\beta_0(\mathcal{L}^-(i))$ increases (Fig. 4.3(b)).

Regular vertices The lower link of a regular vertex at isovalue i is made of one connected component. Then, *(a)* it cannot merge distinct components of $\mathcal{L}^-(i)$ and *(b)* there exists connected paths on $\mathcal{L}^-(i)$ linking it to lower minima. Thus, passing through a regular point does neither *(a)* decrease nor *(b)* increase $\beta_0(\mathcal{L}^-(i))$.

Interior saddles By construction, the lower link of a simple saddle on the interior of $\mathcal{M}$ is made of two connected components (Fig. 3.12(c)). These components can either be linked to *(a)* the same or to *(b)* distinct connected components of $\mathcal{L}^-(i)$. In the latter case *(b)*, $\beta_0(\mathcal{L}^-(i))$ decreases when passing the saddle. In the former case *(a)*, neither $\beta_0(\mathcal{L}^+(i))$ nor $\beta_0(\mathcal{L}^-(i))$ changes ($\beta_0(\mathcal{L}^+(i)) = 1$ for all i below the maximum). However, since f^- is Morse, $f^{-1}(i)$ changes its topology at the saddle. In the interior of surfaces, the only possible topological change of $f^{-1}(i)$ is a change of



	Minima	Removable Saddles	Non-removable Saddles	Maxima
$\Delta\beta_0(L^-(i))$	+1	-1	0	0

Figure 4.3 – *Given a Morse function f^- admitting one maximum and several minima (left inset), the number of connected components of the sub-level set (in blue) increases when passing a minimum (b), decreases when passing a removable saddle (c) and does not change when passing the non-removable saddles (d, e) and the maximum (f).*

$\beta_0(f^{-1}(i))$. Saddles which change $\beta_0(f^{-1}(i))$ while preserving $\beta_0(\mathcal{L}^+(i))$ and $\beta_0(\mathcal{L}^-(i))$ have been shown to correspond to the saddles opening or closing the loops of the Reeb graph of the function (TGSP09) and for surfaces, these loops correspond to the handles of the surface (CMEH*03). Thus, the only interior saddles of f^- for which $\beta_0(\mathcal{L}^-(i))$ does not change are the $2g(\mathcal{M})$ non-removable saddles (Fig. 4.2(b) and Fig. 4.3(d)).

Boundary saddles Simple boundary saddles can be classified into two categories. In the first case (*join* saddles), the lower link is made of two components (each lying on the same boundary component $\mathcal{B} \subseteq \partial\mathcal{M}$) and the upper link of one (Fig. 4.3(e)). The second case (*split* saddles) is symmetric: the lower link is made of one component and the upper link is made of two components on the boundary. Since $\beta_0(\mathcal{L}^+(i)) = 1$ (f^- has only one maximum) and since their lower link is made of only one component, neither $\beta_0(\mathcal{L}^+(i))$ nor $\beta_0(\mathcal{L}^-(i))$ changes when passing split saddles. For a join saddle, noted s_j, the two components of the lower link can either be linked to *(a)* the same or to *(b)* distinct connected components of $\mathcal{L}^-(i)$. In the latter case *(b)*, $\beta_0(\mathcal{L}^-(i))$ decreases when passing s_j. Otherwise *(a)*, neither $\beta_0(\mathcal{L}^-(i))$ nor $\beta_0(\mathcal{L}^+(i))$ changes and there exists a connected path on $\mathcal{L}^-(i)$ connecting the two components of the lower link of s_j. This path encloses the boundary component $\mathcal{B}$ on which s_j lies (Fig. 4.3(e)). This implies that $\{\mathcal{B} - s_j\} \subset \mathcal{L}^-(i)$ since $\mathcal{L}^+(i)$ is made of only one component (the presence of a vertex on $\mathcal{B}$ with a value higher than i would then imply that $\beta_0(\mathcal{L}^+(i)) > 1$). Thus, s_j is the *stratified* global maximum of $f^-_{\mathcal{B}}$. In other words, for each boundary component $\mathcal{B} \subseteq \partial\mathcal{M}$, the only join saddle of f^- for which $\beta_0(\mathcal{L}^-(i))$ does not change is a non-removable saddle (Fig. 4.2).

Maximum The lower link of a maximum is made of only one connected component. Then, a maximum cannot merge or create a new component of $\mathcal{L}^-(i)$. Thus $\beta_0(\mathcal{L}^-(i))$ does not change when passing through a maximum (Fig. 4.3(f)).

The symmetric properties hold for a Morse function $f^+ : \mathcal{M} \to \mathbb{R}$ admitting only one minimum and several maxima. Since the input fields are assumed to be Morse functions, when passing through a given critical point, only one topological event can occur at a time on the level set (Mil63), which enables the viewing of each critical point as an instance of the cases reviewed above. Then, in conclusion, the only critical points of the field for which both $\beta_0(\mathcal{L}^-(i))$ and $\beta_0(\mathcal{L}^+(i))$ do not evolve are the non-removable saddles due to the topology of $\mathcal{M}$; for the removable critical points of f, either $\beta_0(\mathcal{L}^-(i))$ or $\beta_0(\mathcal{L}^+(i))$ changes. Thus, it is possible to constrain the topology of the output field g by only controlling the connectivity of $\mathcal{L}^-(i)$ and $\mathcal{L}^+(i)$. The next subsection presents an algorithm exploiting this property.

4.1.2 Algorithm

Our algorithm naturally follows from the properties discussed in the previous subsection. Given the constraints C_g^0 and C_g^2, it iteratively reconstructs the corresponding sub- and sur-level sets, while removing the optimal set of saddles with regard to the L_∞ norm.

Algorithm description

In the following, we start by describing the algorithm for sub-level set constrained reconstruction.

Sub-level set constrained reconstruction The pseudo-code for sub-level constrained reconstruction is given in Algorithm 1. To guarantee that the input field admits distinct values on each vertex, symbolic perturbation is used, as described in Section 3.1.2. In addition to its scalar value, each vertex v is associated with an integer *offset* (noted $o(v)$) initially set to the actual offset of the vertex in memory. When comparing two vertices (for critical point classification for instance), if these share the same scalar value, their order is disambiguated by their offset o. Algorithm 1 modifies both vertex scalar values and offsets.

Algorithm 1: Sub-level set constrained reconstruction

input : Scalar field $f : \mathcal{M} \to \mathbb{R}$ (with n scalar (f) and offset (o) values);
input : Set of minima constraints to enforce C_g^0;
output: Scalar field $g : \mathcal{M} \to \mathbb{R}$ with enforced minima in C_g^0.

begin
  // T: set of vertices (self-balancing binary search tree).
  $T \leftarrow \emptyset$;
  // i: time (integer) when a vertex was last processed.
  $i \leftarrow 0$;

  // Initialize T with the minima constraints.
  foreach $m \in C_g^0$ **do** $T \leftarrow \{T + m\}$;

  repeat
    $v \leftarrow argmin_{x \in T} \quad f(x)$;
    $T \leftarrow \{T - \{v\}\}$;
    mark v as **visited**;
    // Add unvisited neighbors.
    $T \leftarrow \{T \cup \{v_n \in Lk(v) \mid v_n$ is not **visited**$\}\}$;
    $A[i] \leftarrow v$;
    $i \leftarrow i + 1$;
  until $T = \emptyset$;

  // Scalar and offset value update, for all the vertices.
  // Make the ordering on g (scalar and offset values) consistent with the order of visit.
  for $j \leftarrow 0$ ***to*** n **do**
    if $j \neq 0 \quad \&\& \quad f(A[j]) < g(A[j-1])$ **then**
      $g(A[j]) \leftarrow g(A[j-1])$;
    else
      $g(A[j]) \leftarrow f(A[j])$;
    $o(A[j]) \leftarrow j$;
end

The algorithm starts by pushing the minima constraints C_g^0 into a self-balancing binary search tree (noted T) ordered by scalar value and offset. Then, the sub-level sets are iteratively reconstructed, o), one vertex at a time, in a flooding fashion: the unvisited neighbors of the visited vertex are added to T and the vertices of T are uniquely visited in increasing order of scalar value (and offset) until the entire domain is processed. For instance, Fig 4.4 shows the removal of the lowest minimum of f. Hence, all the minima of f have been added to C_g^0 except for the global minimum. The corresponding flooding is progressively shown in the middle of Fig. 4.4, where the visited and unvisited vertices appear in blue and green respectively. The resulting order of visit of each vertex is stored in the array A. The last step of the algorithm traverses A and updates the vertex scalar values and offsets such that the order defined by the output field is equivalent to the order of visit of the vertices (then the sub-level sets $\mathcal{L}^-(i)$ of the output field indeed correspond to the iteratively reconstructed sub-level sets). As shown in Fig. 4.4 (right), this strategy for function value update has the effect of flattening the output in the vicinity of the removed minima.

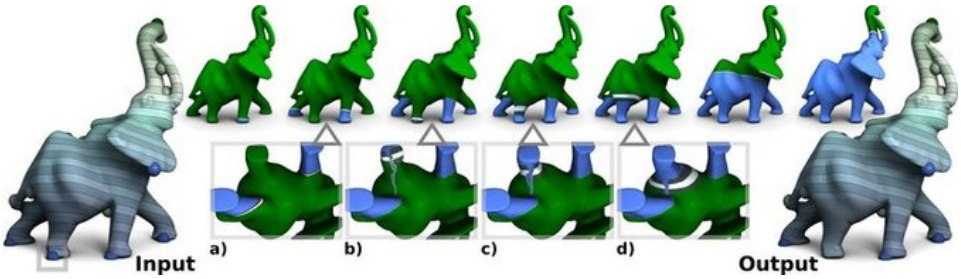


Figure 4.4 – *Removing the lowest minimum (small box on the input) by sub-level set constrained reconstruction (in blue). The algorithm enforces the minima constraints while implicitly removing saddles.*

Since the sub-level sets are grown by adding the vertices of T with smallest function value first, if a connected component of $\mathcal{L}^-(i)$ were to hit a minimum constraint $m \in C_g^0$ before the latter was popped out of T, this would imply that m had a neighbor with lower initial function value, through which the component entered the link of m. This implies that m was not a minimum in the input, which is an invalid constraint. Then, for each $m \in C_g^0$, m is visited before the vertices of its link; hence the vertices of C_g^0 are all minima in the output. The other vertices which do not belong to C_g^0 can only be visited by the iterative growth of $\mathcal{L}^-(i)$, after that one of the vertices of their link has been visited. Hence, their lower link is not empty in the output and the vertices $m \in C_g^0$ are then the only minima of the output (Fig. 4.4).

When passing a saddle s of the input which used to join distinct components of sub-level sets, if only one component of $\mathcal{L}^-(i)$ hits the saddle, this implies that the minimum which created initially the other component does not belong to the constraints C_g^0. Then, the component of $\mathcal{L}^-(i)$ traverses s and continues to grow by visiting vertices with smallest values first, eventually sweeping the removed minimum (Fig. 4.4(b)-(d)). Otherwise, s is maintained as a saddle in the output.

Hence, the algorithm implicitly removes saddles given the extrema constraints and guarantees a valid topology of the output. Note that, since the algorithm visits the vertices of T with smallest value first, the saddle removed with one minimum m is the lowest saddle s which used to join the sub-level set component created by m in the input (i.e. the next saddle from m up the join tree): thus the algorithm minimizes $|f(m) - f(s)|$ when removing one saddle s along with one minimum m. Since the update of function value will lift m up to the level of s ($g(m) \leftarrow f(s)$, algorithm 1), $||f - g||_\infty$ will be equal to $|f(m) - f(s)|$ (for instance, in Fig. 4.5, B is lifted

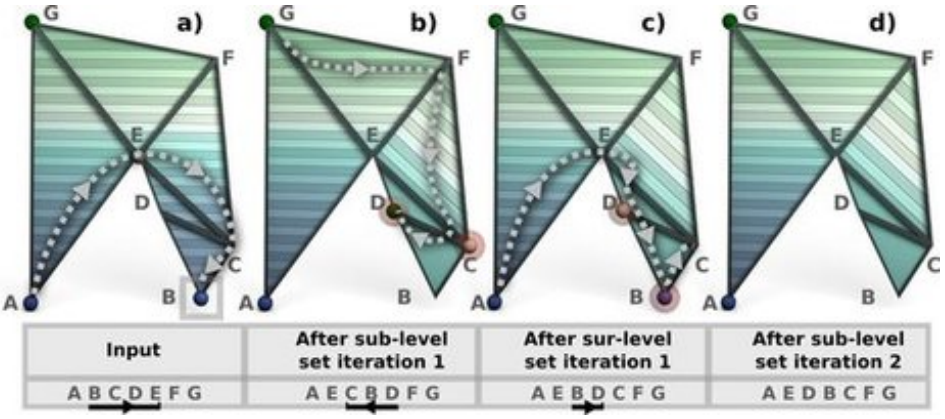


Figure 4.5 – *Sub-level set constrained reconstruction can introduce residual maxima (red spheres): in (a), all the neighbors of D are visited before it, hence yielding a maximum (b). Symmetrically, in (b), all the neighbors of B are visited before it, yielding a minimum (c). Alternating sub- and sur- level set reconstruction reduces the (offset) function difference between the residual extrema and their corresponding saddle (cf. vertex ordering, bottom), and converges to the removal of all the residuals.*

up to the level of E). Thus the algorithm removes the optimal set of saddles with regard to $||f - g||_\infty$, hence guaranteeing a small distance between the input and the output (for the regions where no simplification is needed, the function is unaltered).

Sur-level set constrained reconstruction The constraints C_g^2 are enforced with the symmetric algorithm: the vertices of C_g^2 are initially pushed in T and the vertices of T are visited in decreasing order of function value (the update of the function values is also symmetric).

Overall algorithm As shown in Fig. 4.5, while algorithm 1 guarantees that the constraints C_g^0 will be the only minima of the output, it does not guarantee that C_g^2 will be the only maxima. When the reconstructed sub-level set removes a saddle, the algorithm visits the vertices of lowest function value in priority, possibly leaving islands of non-visited vertices behind (Fig. 4.5(a)), yielding residual maxima in the output function (Fig. 4.5(b)). The symmetric remark goes for the sur-level set reconstruction regarding minima constraints. To remove these residuals, our overall algorithm successively alternates sub- and sur-level set reconstructions until C_g^0 and C_g^2 are the only extrema. We show in the following that this process converges.

Convergence Algorithm 1 lifts up the minima to remove, since they are visited after their associated saddle (Fig. 4.5(a)). Then, when removing a

minimum m (B, Fig. 4.5(a)), residual critical points can only occur higher than the minimum's associated saddle (E, Fig. 4.5(a)), but lower than its next vertices in the global vertex ordering (F and G, Fig. 4.5(a)). Symmetrically, sur-level set reconstruction pushes down the maxima to remove. Then new residual critical points (B and D, Fig. 4.5(c)) occur lower than the residual saddle of the previous step (C, Fig. 4.5(b)), but still higher than the original (E, Fig. 4.5(a)). Alternating sub- and sur-level set reconstruction will keep on reducing the function range where the residual extremum and its corresponding saddle appear. Eventually these will be consecutive in the global vertex ordering (Fig. 4.5, bottom) leaving no more room for new residuals at the next iteration.

From symbolic to numerical perturbation After convergence, it may be useful to convert the symbolic perturbation (vertex offset $o(v)$) into numerical perturbation, to represent the output field g with a numerical value only for each vertex. The final array A (algorithm 1) is traversed in increasing order and whenever a vertex is at the same value (or lower) than its predecessor ($g(A[i]) \leq g(A[i-1])$), its function value is increased by an arbitrarily small value ξ: $g(A[i]) \leftarrow g(A[i-1]) + \xi$. This numerical perturbation should be restricted only where it is needed (flat regions of g) to maintain $||f-g||_\infty$ to a small value. For instance, in Fig. 4.5, the vertices D, B and C should all have a final function value in the interval $(f(E), f(F))$. Hence, ξ should be smaller than $\frac{\delta_f}{n}$, where δ_f is the smallest (non-zero) function value absolute difference in the input and n is the number of vertices in $\mathcal{M}$.

Algorithm properties

Relation to ϵ-simplification The implicit pairing performed by our algorithm is compatible with the pairing of critical points based on persistence: given one extremum removal, it pairs a minimum (respectively, a maximum), with its closest saddle up the join tree (respectively, down the split tree). Moreover, given one extremum removal, $||f-g||_\infty$ will be equal to the absolute difference in function value between the extremum and its paired saddle. For instance in Fig. 4.5, B is paired with E and $||f-g||_\infty$ is equal to $|f(B)-f(E)|$. Thus, if the input constraints are selected according to topological persistence (the persistence of the pairs associated with each critical point of C_g^0 and C_g^2 is higher than ϵ), then $||f-g||_\infty \leq \epsilon$.

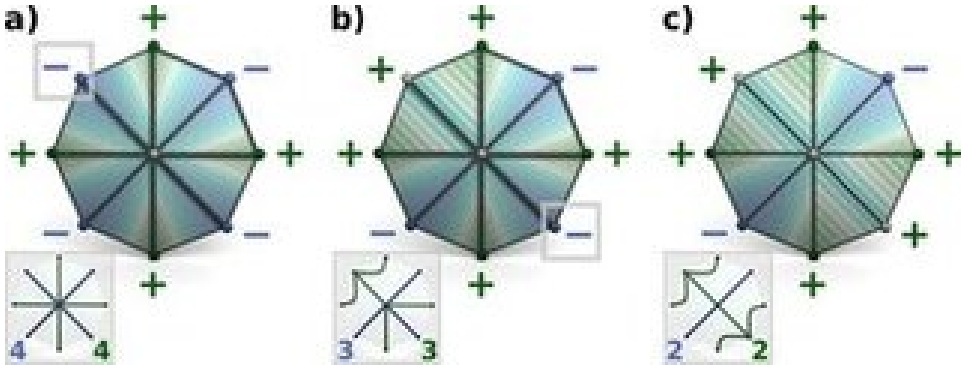


Figure 4.6 – *Simplifying a saddle with high multiplicity (a): 4 components of sub- and sur-level sets merge in the saddle (inset Reeb graph: sub- and sur- level sets are marked in blue and green respectively). Removing one extremum (box in (a)) decreases the number of components of the lower and upper links by 1 (b). Removing other extrema (box in (b)) eventually decreases this number to 2, yielding a simple saddle (c).*

Non-Morse inputs Multi-saddles may occur in the input, preventing f from being a Morse function. For these, the lower and upper links can be made of more than two components. Our algorithm handles these degenerate cases with no modification: removing one extremum associated with a multi-saddle will simply decrease the saddle's multiplicity in the output by one (Fig. 4.6).

4.1.3 Results and discussion

In this section, we present practical results of our algorithm obtained with a C++ implementation on a computer with an i7 CPU (2.93 GHz).

Time requirement

The algorithm uses no computationally expensive topological abstraction, such as a Morse-Smale complex or even a contour tree. Therefore each iteration is extremely fast in practice. Given a surface with n vertices, inserting and removing a vertex from the self-balancing binary search tree T takes $O(log(n))$ time. Each vertex is uniquely visited. Thus, the complexity of an iteration is $O(n\ log(n))$, irrespectively of the number of critical points to remove. In theory, the number of iterations required for the algorithm to converge could possibly be non-negligible. Given one extremum removal, after each reconstruction, the distance in the global vertex ordering which separates a new residual extremum from its corresponding saddle decreases at least by one (this distance is illustrated by a black arrow in Fig. 4.5, bottom). As this distance can initially be close to the number of vertices in the mesh, n reconstructions could be required, yielding

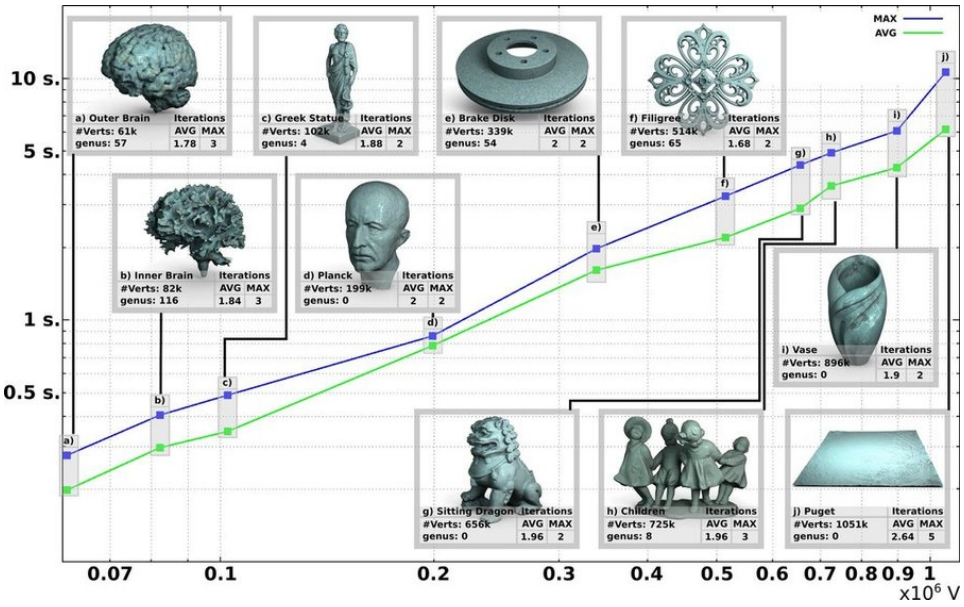


Figure 4.7 – *Running times (log scale) for the simplification of random functions, with a random constraint selection (C_g^0, C_g^2), 50 runs per data-set.*

an $O(n^2\ log(n))$ worst case complexity, irrespectively of the number of removed extrema.

Traditionally, random scalar fields are considered as relevant approximations of worst-case scenarios. Moreover, considering multiple instances increases the chances to get a proper worst-case approximation. We show in Fig. 4.7 the average and maximum running times (in log scale) for the algorithm to achieve convergence on a set of meshes (including examples with high genus, up to 116), for which 50 instances of random fields have been considered and for which the constraints C_g^0 and C_g^2 are random subsets of the fields' extrema. In particular, critical points from C_f^0 are uniquely added to C_g^0 in random order until $|C_g^0|$ is equal to a random fraction of $|C_f^0|$ (the constraint set C_g^2 is constructed similarly). For most data-sets, the average number of required iterations is smaller than or equal to 2 and the maximum number of iterations is never greater than 5. This shows that, from a practical point of view, the number of required iterations is negligible with regard to n, hence yielding $O(n\ log(n))$ practical running time. In our experiments, the algorithm took at most 10.7 seconds to compute on a mesh with 1 million vertices (6.3 million simplices total).

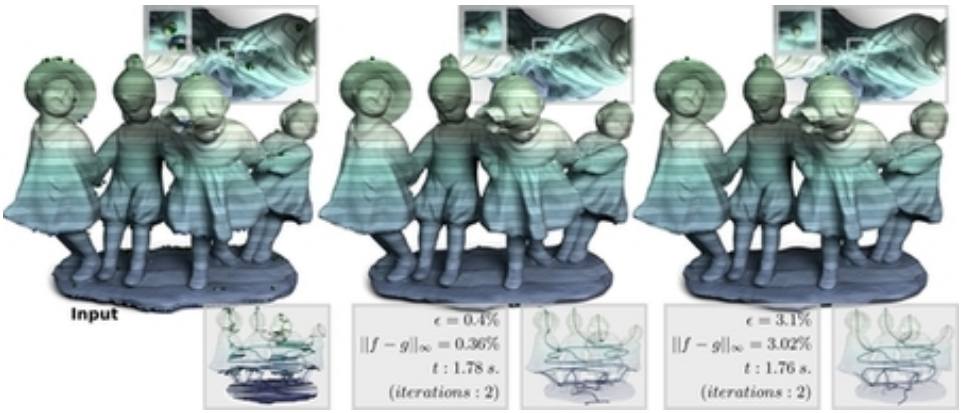


Figure 4.8 – *In the special case where the critical points in C_g^0 and C_g^2 are all more persistent than ϵ, our algorithm produces an ϵ-simplification ($||f - g||_\infty \leq \epsilon$). Both the (vertex) position and the function value of the remaining removable critical points are preserved after simplification (top insets), even in the presence of multi-saddles (6 in the input). The topology of the resulting field is summarized with the inset Reeb graph for illustration purpose (input surface: 725k vertices).*

Discussion

A unique aspect of our algorithm is its ability, given the constraints C_g^0 and C_g^2, to automatically identify and remove the optimal set of saddles with regard to the L_∞ norm. Moreover, this is accomplished without the need to carry a union-find data-structure unlike previous techniques. Although this data-structure has nearly linear time complexity in theory, practically it could cause slowdowns by a non-negligible constant factor, given the algorithm's low resource requirement. Also, after simplification, the (vertex) position of the remaining critical points is preserved. In the special case where the critical points of C_g^0 and C_g^2 are selected based on topological persistence, the algorithm produces a standard ϵ-simplification, as shown in Fig. 4.8. In contrast to previous approaches, our algorithm directly works on a PL representation of the field, which is more acceptable application-wise. Importantly, it is also more general as critical points can be removed arbitrarily (at the exception of the non-removable critical points summarized in Fig. 4.2), irrespectively of the employed feature-selection strategy.

Limitations

Given our formulation of general simplification, for specific constraint configurations, the value of the remaining critical points may change after simplification. For instance, if the lowest minimum in C_g^0 is initially higher than the highest maximum in C_g^2, the algorithm will change their

values to satisfy the topological constraints. Also, when removing only one extremum from the boundary, it will be replaced by a boundary saddle if it is associated by the algorithm (with regard to the L_∞ norm) with an interior saddle (Fig. 4.6). This is due to the fact that the number of critical points must be even on each boundary component, which may be counter-intuitive from a user's perspective. Moreover, whereas our algorithm removes the optimal set of saddles with regard to $||f-g||_\infty$ given some extrema constraints, our strategy for function value update (which is purely based on *flooding*) does not guarantee a minimization of $||f-g||_\infty$, although the resulting L_∞ norm is close to the theoretical minimum. In the context of persistence driven simplification, Bauer et al. (BLW12) showed that optimality could be reached by a combination of *carving* (i.e. pulling the saddles halfway towards their associated extremum) and *flooding* (i.e. pushing the extrema halfway towards their associated saddle) at the expense of no longer guaranteeing a function value lock on the maintained critical points. By simplifying all the pairs less persistent than ϵ, their approach yields $||f-g||_\infty \leq \frac{\epsilon}{2}$. In contrast, like in (EMP06), our approach yields $||f-g||_\infty \leq \epsilon$ but locks the maintained critical points in terms of function value. In the context of general simplification, the optimal balance between carving and flooding is more difficult to evaluate, as it is no longer a local decision which depends only on the pair of removed critical points (excessive carving could break the enforcement of near-by extrema located in the vicinity of the removed saddles). We addressed this issue and provided an optimal algorithm, derived from this approach but slower in practice and more difficult to implement, in a follow-up work (TGP14). Like any combinatorial approach, our algorithm provides strong guarantees on the topology of the output at the expense of its geometrical smoothness. Unlike previous combinatorial algorithms using *carving* (EMP06, AGH*09, BLW12), our algorithm uses *flooding* only. Hence, it does not suffer from the usual artifacts of carving (visible thin paths linking sets of removed critical points), but from those of flooding (flat regions in the output, Fig. 4.4, right). In our experiments, we found that these artifacts were usually little noticeable, unless the features removed by the user span a large region of the domain (Fig. 4.4, right). If smoothness is desired, our approach can be combined with a numerical technique to provide smooth outputs that still benefit from the topological guarantees of our algorithm (see Sec. 7.1).

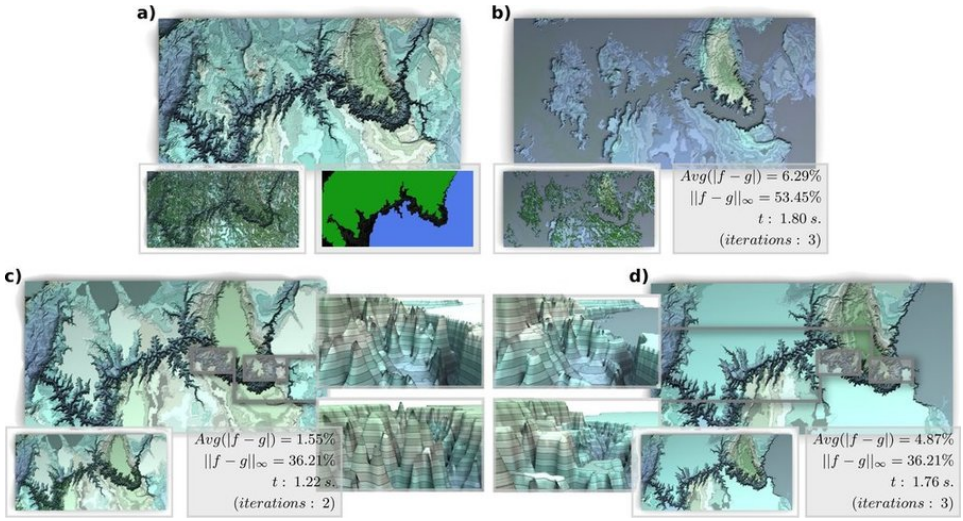


Figure 4.9 – *Simplifying the Grand Canyon ((a), 500k vertices, 65,526 critical points, bottom) with a location driven feature selection (black: canyon, green: north rim, blue: south rim). (b) Maintaining only one minimum and removing all the critical points from the canyon (16,756 critical points remain) emphasizes the topological features of the rims and simulates a massive flooding of the canyon. (c) Removing all the critical points from the rims (2,296 remaining) emphasizes the topological features inside the canyon. An ϵ-simplification with compatible L_∞ norm (d) completely discards the features irrespectively of their location (zoom insets) while yielding a worse average data fitting ($Avg(|f-g|)$).*

Application

In practice, non-relevant critical points can often be easily discriminated, by discarding either certain range or domain ranges. We illustrate this process with terrain simplification, where we show that ad-hoc application driven constraints can be easily incorporated in our simplification scheme to ease further analysis. This is particularly useful if this latter analysis involves the computation of topological abstractions (for segmentation tasks for instance). Then, we show that a pre-simplification of the data based on application driven constraints can greatly accelerate the computation of topological abstractions. Other application examples of this technique are demonstrated further down this manuscript.

Topological simplification of terrains can be particularly useful for topographic analysis or water flow simulations as discussed by Bremer et al. (BEHP04). However, the original topological persistence measure can be unsatisfactory for selecting features in such a context, as the characterization of features of interest is application dependent and at times can also be data-set dependent. Fig. 4.9 exemplifies this observation on the Grand Canyon elevation data-set, which initially counts 65,526 critical points. The Grand Canyon can be decomposed in three major re-

Data-set	$\lvert C_g \rvert$	Simplification Time (s.)	Reeb Graph Time (s.)	Speedup
Original - Fig. 4.9(a)	65,526	-	125.723	-
Rims - Fig. 4.9(b)	16,756	1.783	12.976	**9.69**
Canyon - Fig. 4.9(c)	2,296	1.123	4.432	**28.37**
Persistence - Fig. 4.9(d)	12	1.702	0.313	**402.28**

Table 4.1 – *Computation times and speedups for the Reeb graph computation of the simplified data, with the algorithm by Patane et al. (PSF08).*

gions from a geographic point of view: the canyon itself, its north rim and its south rim. These regions (in black, green and blue respectively in Fig. 4.9(a)) have been initially extracted by segmenting the image along high elevation gradient and interactively completed by the user. Based on this initial decomposition, we present two simplification scenarios. First, in Fig. 4.9(b), the topology of only the rims has been emphasized: only the lowest minimum above 53% of the total elevation difference has been maintained, the critical points inside the canyon have been selected for removal and all the maxima on the rims above 53% of elevation have been maintained. In less than 2 seconds, our algorithm constructs the corresponding *flooded* Grand Canyon, while enforcing the preservation of the selected topological features on the rims. A contrary scenario would consist in emphasizing the peaks inside the canyon. For instance, the result of such a simplification strategy can drive a mesh simplification procedure for an interactive fly-through within the canyon. In that case (Fig. 4.9(c)), only the global minimum has been maintained and all the maxima outside of the canyon have been selected for removal. Note that in comparison, a standard ϵ-simplification with compatible L_∞ norm (Fig. 4.9(d)) is unsatisfactory as it completely discards the topological features irrespectively of their location (zooms in Fig. 4.9), while yielding a worse average data-fitting ($Avg(|f-g|)$).

Table 4.1 shows the computation times for the Reeb graph construction of the simplified data, with the algorithm by Patane et al. (PSF08) (as described in the previous chapter, the construction of this topological abstraction is instrumental for data segmentation for instance). This table shows that the simplification pre-processing step usually takes much less time than the Reeb graph computation itself, while considerably improving its own performances with up to two orders of magnitude speedups (depending on the number of critical points in g).

Concluding remarks

This section presented a combinatorial approach for the general simplification of piecewise linear scalar fields on surfaces. By abstracting the approach from the concepts of persistent homology, we believe to have presented a simpler, more intuitive and more general description of scalar field topological simplification. Also, we enumerated all the configurations for which critical points were *non-removable* given the topology of the domain, for surfaces with or with out boundary. From this, we derived a strategy that allows for the arbitrary suppression of the *removable* critical points of the field. We presented a simple iterative algorithm for general topological simplification which, given some constraints on the extrema of the output field, implicitly identifies and removes the optimal set of saddles with regard to the L_∞ norm, hence guaranteeing a small distance $||f - g||_\infty$. Although it is iterative, extensive experiments on approximated worst-case scenarios showed that the algorithm rarely takes more than 2 iterations to converge. The algorithm uses no computationally expensive topological abstraction, such as a Morse-Smale complex or even a contour tree; hence it is very fast in practice. We demonstrated that it could be used as a pre-processing step to accelerate topological abstraction computation algorithms. In contrast to previous combinatorial approaches, our approach works directly on a PL representation of the field, is robust against multi-saddles, and handles surfaces with or without boundary. Moreover, our approach solves a more general problem, for which ϵ-simplifications have been shown to be a special case.

Thanks to its generality, robustness, good time complexity, ease of implementation and practical performances, we consider this algorithm as the reference for the topological simplification of scalar data on surfaces.

A natural direction for future work is the extension of this approach to volumetric data-sets. As the algorithm works on the 0 and 1-simplices of the domain only, it can be used in principle directly for domains of arbitrary dimension. However, as the dimension of the domain increases, new types of saddle points appear and more subtle topological transitions occur on the level sets (genus changes through 1-2-saddle pairs). Hence, enforcing the connectivity of the sub- and sur-level sets is insufficient for the removal of 1-2-saddle pairs; the genus of the iso-surfaces also needs to be efficiently controlled. However, this problem is NP-hard as recently shown by Attali et al. (ABD*13). This indicates that the design of a practical algorithm with strong topological guarantees is challenging.

4.2 Efficient Reeb graph computation for volumetric meshes

As discussed in the previous chapter, the Reeb graph is a fundamental data-structure in scientific visualization for contour indexing with applications in fast contour handling, data segmentation and feature extraction. However, as detailed in Section 3.3, algorithms with strong quadratic behavior in practice were only available for volumetric meshes and were therefore not practically applicable for use in interactive applications.

In this section, I present a practical algorithm to compute Reeb graphs on volumetric meshes in $\mathbb{R}^3$ (in particular tetrahedral meshes) that runs in practice with comparable efficiency to a contour-tree algorithm, enabling the practical generalization of contour tree based visualization techniques to meshes of arbitrary topology. Our approach is based on the key concept of *loop surgery*, inspired from surgery theory (Wal70). In particular, we transform the input domain by a sequence of symbolic cuts such that the Reeb graph of the input scalar field defined on the transformed domain is guaranteed to be loop free, and hence computable with efficient contour tree algorithms. Then, some *inverse symbolic cuts* are performed in the topology domain to convert the computed contour tree into the Reeb graph of the original scalar field. We show that these *symbolic cuts* can be computed in an efficient manner, with reasonable computation overhead with respect to contour tree computation. Extensive experiments show that our approach improves state-of-the-art techniques for computing Reeb graphs by several orders of magnitude in terms of running time, with reasonable memory footprint.

This work makes the following contributions:

1. A procedure called *loop surgery* to reduce the problem of computing a Reeb graph to that of a contour tree. We believe this is an important result, since the join-split algorithm (CSA00) for computing contour trees is well known to have not only optimal theoretical complexity, but also simple and practical implementation.

2. A practical algorithm for computing Reeb graphs with complexity $O(n \log n + N\alpha(N) + g(\mathcal{M}_\partial) \times N_S)$. For practical examples, g, which is equal to the number of handles of the domain, is a small constant, and systematic experiments show a speedup over previous algorithms by several orders of magnitude on average.

3. A proof showing necessary and sufficient conditions for a loop free

Reeb graph to be computed correctly by the join-split contour tree algorithm.

4.2.1 Preliminaries

In the following, we consider a PL Morse scalar field $f : \mathcal{M} \to \mathbb{R}$ defined on a PL 3-manifold $\mathcal{M}$ embedded in $\mathbb{R}^3$.

Loops in Reeb graphs on PL 3-manifolds in $\mathbb{R}^3$

The key idea of *loop surgery* is to define a sequence of operations that transform $\mathcal{M}$ to $\mathcal{M}'$ with $f' : \mathcal{M}' \to \mathbb{R}$ valued by f such that $\mathcal{R}(f')$ becomes loop-free, and then efficiently computable with contour tree algorithms. Therefore, we carefully characterize the loops (independent cycles) of $\mathcal{R}(f)$ prior to introducing loop surgery.

Since $\mathcal{M}$ is compact and embeddable in $\mathbb{R}^3$, $\partial\mathcal{M}$ is necessarily non-empty, orientable and closed (DG98) but possibly disconnected.

Let f_∂ be the restriction of f to $\partial\mathcal{M}$. We will first assume that both f and f_∂ are PL Morse functions (degenerate cases will be discussed later). Let $\mathcal{R}(f_\partial)$ be the Reeb graph of f_∂. Then, we have the following relation (CMEH*03):

$$l(\mathcal{R}(f_\partial)) = g(\partial\mathcal{M}) \tag{4.4}$$

where $g(\partial\mathcal{M})$ is the sum of the genuses of the boundary components of $\mathcal{M}$. The key property that will allow us to implement loop surgery in an efficient manner is the fact than the topology of $\mathcal{M}$ is closely related to that of $\partial\mathcal{M}$. In particular, the number of handles of $\mathcal{M}$ is the first Betti number, $\beta_1(\mathcal{M})$, which is given by the following relation (DG98):

$$\beta_1(\mathcal{M}) = g(\partial\mathcal{M}) \tag{4.5}$$

In simpler words, this relation expresses the fact that each handle of the volume $\mathcal{M}$ corresponds to a tunnel of its boundary surface.

As discussed in (CMEH*03) in any dimension the construction of the Reeb graph can lead to the removal of 1-cycles, but not the creation new ones. Therefore, the number of loops of $\mathcal{R}(f)$ cannot be greater than the first Betti number of $\mathcal{M}$:

$$l(\mathcal{R}(f)) \leq \beta_1(\mathcal{M}) \tag{4.6}$$

In conclusion, the number of loops of $\mathcal{R}(f)$ cannot be greater than the number of loops of $\mathcal{R}(f_\partial)$:

$$l(\mathcal{R}(f)) \leq \beta_1(\mathcal{M}) = g(\partial\mathcal{M}) = l(\mathcal{R}(f_\partial)) \tag{4.7}$$

(a) $\mathcal{M}, f$ (b) $\mathcal{R}(f_\partial)$ (c) Cutting surfaces (d) $\mathcal{M}', f'$ (e) $\mathcal{R}(f')$

Figure 4.10 – *Overview of Reeb graph computation based on loop surgery: The input defines a scalar function f (color gradient) on $\mathcal{M}$ (a). The Reeb graph of the function restricted to the boundary $\mathcal{R}(f_\partial)$ (b) is used to identify* loop saddles *(blue squares).* Cutting surfaces *(c) of each loop saddle are used to transform the domain (d) to $\mathcal{M}'$. The Reeb graph $\mathcal{R}(f')$ is loop free (e).* Inverse cuts *are applied to circled critical point pairs to obtain the Reeb graph $\mathcal{R}(f)$ of the input function.*

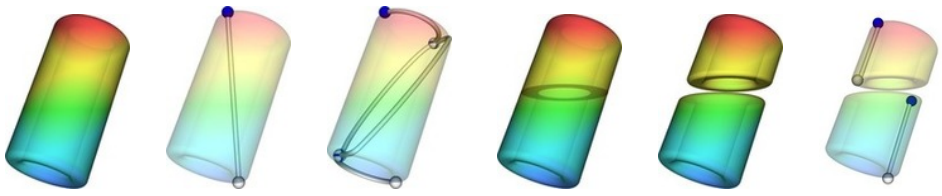
A direct extension of the equation 4.7 is the following property:

Property 17 (Existence of loops) *The existence of loops in $\mathcal{R}(f)$ implies the existence of corresponding tunnels in both $\mathcal{M}$ and $\partial\mathcal{M}$, and thus of corresponding loops in $\mathcal{R}(f_\partial)$. The inverse is not necessarily true.*

A consequence of this property is that one can deduce information about the loops of $\mathcal{R}(f)$ by just studying $\mathcal{R}(f_\partial)$. In particular, a loop of $\mathcal{R}(f_\partial)$ will yield a loop in $\mathcal{R}(f)$ if its preimage through ϕ_∂ at a given isovalue contains contours of f_∂ that form the boundary components of distinct contours of f. It follows that each loop $\mathcal{L}$ of $\mathcal{R}(f)$ maps through ψ to an interval $\psi(\mathcal{L}) \subseteq \psi_\partial(\mathcal{L}_\partial)$ being a subset of the image by ψ_∂ of a distinct loop $\mathcal{L}_\partial$ of $\mathcal{R}(f_\partial)$. It follows that the loop saddles of each loop $\mathcal{L} \in \mathcal{R}(f)$ lie on a monotone path on $\mathcal{J}(f)$ (or $\mathcal{S}(f)$), being a subset of the monotone path linking the loop saddles of the corresponding loop $\mathcal{L}_\partial \in \mathcal{R}(f_\partial)$. This latter observation will be instrumental in our algorithm.

Loop surgery

When f_∂ is PL Morse, there exists a unique pair of loop saddles for each loop, consisting of an "opening" split saddle and a "closing" join saddle. The existence of such pairs is guaranteed by extended persistence (AEHW04). We uniquely associate each loop of $\mathcal{R}(f_\partial)$ with the closing saddle of this pair. Moreover, by property 17, it is possible to associate each loop in $\mathcal{R}(f)$ with the loop saddles of the corresponding loop in $\mathcal{R}(f_\partial)$. Notice that some loop saddles of $\mathcal{R}(f_\partial)$ may not be associated



(a) $\mathcal{M}, f$ (b) $\mathcal{R}(f)$ (c) $\mathcal{R}(f_\partial)$ (d) S_i (e) $\mathcal{M}', f'$ (f) $\mathcal{R}(f')$

Figure 4.11 – *Example where the domain (a) is not simply connected and $\mathcal{R}(f)$ is loop free (b). The Reeb graph of the boundary, $\mathcal{R}(f_\partial)$, has a loop (c), therefore loop surgery is performed (d), (e). Even after loop surgery, each component of $\mathcal{M}'$ is not necessarily simply connected, but the Reeb graph of each component is loop free (f).*

with any loop of $\mathcal{R}(f)$. We will describe in the algorithm section how to identify candidates for the loop saddles of f given those of f_∂.

Loop surgery consists of transforming the domain $\mathcal{M}$ such that $\mathcal{R}(f)$ becomes loop-free. In other words, loop surgery breaks the loops of $\mathcal{R}(f)$ and reflects that transformation on $\mathcal{M}$. For each loop 1-saddle s_i of f with value $f(s_i)$, we define its *cutting surface* S_i as a contour of f (a connected component of isosurface inside the volume) at value $f(s_i) - \epsilon$ such that $f(s_i) - \epsilon$ is a regular value of f, there is no critical value in $[f(s_i) - \epsilon, f(s_i))$ and S_i intersects one of the connected components of the lower star $St^-(s_i)$ in the volume. The symbolic cuts transforming $\mathcal{M}$ into $\mathcal{M}'$ consists of cutting $\mathcal{M}$ along each defined cutting surface, as illustrated in figures 4.10(c) and 4.10(d).

On $\mathcal{R}(f)$, cutting along a cutting surface at a regular value is equivalent to cutting a 1-simplex of $\mathcal{R}(f)$ and creating a new pair of critical nodes (a minimum and a maximum, as illustrated in figure 4.10(e)). Then, the Reeb graph of the function $f' : \mathcal{M}' \to \mathbb{R}$, f' being the function valued by f after symbolic cuts, is guaranteed to be loop free: all the possible loops have indeed been broken, as shown in figure 4.10(e).

Once the loop-free Reeb graph $\mathcal{R}(f')$ is computed, *inverse cuts* can be applied in a straightforward manner by removing pairs of minimum and maximum nodes generated by the same cutting surface, and gluing together the corresponding 1-simplices.

Loop free Reeb graph computation

The algorithm presented by Carr et al. (CSA00) computes the contour tree by tracking the joining of sub- and sur-level set components. Traditionally, simply-connectedness of $\mathcal{M}$ has been used as a condition to ensure correctness of this algorithm. However, a Reeb graph can be loop free even

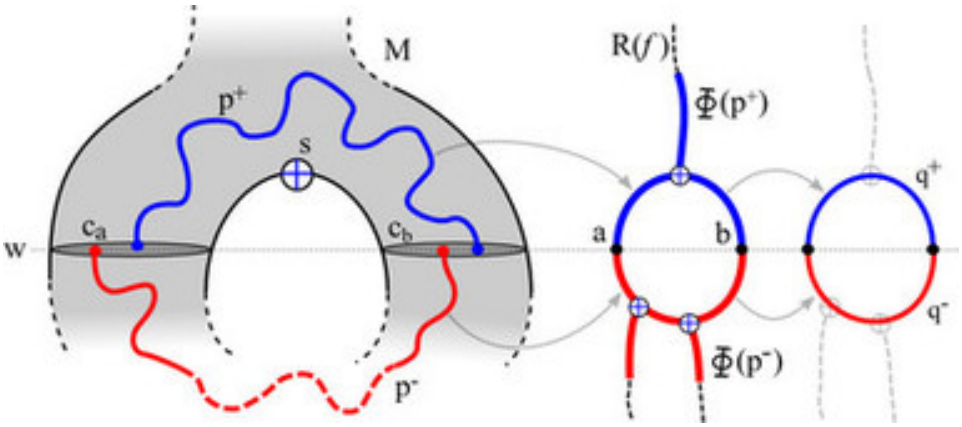


Figure 4.12 – *Reference figure for property 18. Paths forming a loop in $\mathcal{M}$ map to connected components forming a loop in $\mathcal{R}(f)$.*

when the domain is not simply connected, as shown in figure 4.11(b). This is especially important, since our loop surgery procedure does not guarantee that the domain is divided into simply connected regions. Therefore, we prove necessary and sufficient conditions for this contour tree algorithm to work. The following property shows when a saddle creates a loop in the Reeb graph.

Property 18 (Loop saddles) *Let $f : \mathcal{M} \to \mathbb{R}$ be a PL Morse scalar field and σ be a valence-3 0-simplex of $\mathcal{R}(f)$ being the image by ϕ of a join saddle s (see figure 4.12). If the contours joined by the saddle are on the boundary of the same sub-level set component, then there exists a loop in $\mathcal{R}(f)$ for which σ is the highest loop saddle.*

Proof. Refer to figure 4.12 in the following. Given σ, s, f, and $\mathcal{R}(f)$, let ϵ be a small number such that there is no critical value in the range $[f(s) - \epsilon, f(s))$. The existence of such an epsilon is guaranteed because the critical values of a Morse function are distinct. Let $w = f(s) - \epsilon$, and a and b be the points on the two downward 1-simplices of $\mathcal{R}(f)$ from σ such that the value of their corresponding contours c_a and c_b is w, *i.e.*, $f(c_a) = f(c_b) = w$. By construction, s is a join saddle, therefore there exists a path p^+ in $\mathcal{M}$ connecting a point in c_a with a point in c_b such that all its interior is in the sur-level set $\mathcal{L}^+(w)$. ϕ is continuous, therefore a connected component p^+ in $\mathcal{M}$ is mapped to a connected component $\phi(p^+)$ in $\mathcal{R}(f)$, such that a and b are connected in $\mathcal{R}(f)$ by a path $q^+ \subseteq \phi(p^+)$ whose interior is strictly above w. Since by hypothesis c_a and c_b are on the boundary of the same sub-level set component, there exists also a path p^- in $\mathcal{M}$ connecting a point in c_a with a point in c_b such that all its interior is in the sub-level set $\mathcal{L}^-(w)$. Therefore a and b are also connected in $\mathcal{R}(f)$ by a path $q^- \in \phi(p^-)$ whose interior is strictly below w. The two paths $q^+ + q^-$ form a loop. □

By applying property 18, we prove necessary and sufficient conditions for computing correct loop free Reeb graphs using tracking of sub- and sur-level sets.

Property 19 (Loop free Reeb graphs) *Let $f : \mathcal{M} \to \mathbb{R}$ be a PL Morse scalar field. Its Reeb graph $\mathcal{R}(f)$ is loop free if and only if every valence-3 0-simplex of $\mathcal{R}(f)$ is the image through ϕ of a saddle of f where distinct components of sub- or sur-level sets join.*

Proof. First we prove that if a Reeb graph $\mathcal{R}(f)$ is loop free, every valence-3 0-simplex σ is the image through ϕ of a saddle s of f that joins distinct components of sub- or sur-level sets. Assume that there exists a valence-3 0-simplex that joins contours that are on the boundary of the same sub- or sur-level set component. Then by property 18, there must be a loop in $\mathcal{R}(f)$. This contradicts the hypothesis statement that $\mathcal{R}(f)$ is loop free.

Next, we prove that if every valence-3 0-simplex of $\mathcal{R}(f)$ being the image through ϕ of a saddle joins distinct sub- or sur-level set components, then $\mathcal{R}(f)$ is loop free. Assume for contradiction that $\mathcal{R}(f)$ has a loop, s is the highest join saddle in the loop, and σ its image through ϕ. Pick ϵ, w, a, b, c_a, and c_b as before. The paths q^+ and q^- exist in $\mathcal{R}(f)$ connecting a and b in a loop. The inverse map ϕ^{-1} is a continuous mapping of points of the Reeb graph to contours of $\mathcal{M}$, therefore $\phi^{-1}(q^-)$ is a connected component in $\mathcal{M}$, connecting c_a to c_b in $L^-(w)$. Therefore, c_a and c_b are on the boundary of the same connected component of $\mathcal{L}^-(w)$, which contradicts the hypothesis. The same argument holds for split saddles. □

Property 19 shows that we can use the contour tree algorithm presented by Carr et al. (CSA00) to compute loop free Reeb graphs, by tracking the connectivity evolution of sub- and sur- level sets. Performing loop surgery guarantees that $\mathcal{R}(f')$ is loop free, and then computable with the contour tree algorithm.

4.2.2 Algorithm

Our strategy for Reeb graph computation is summarized in Algorithm 2. First, the domain is symbolically cut to ensure that the Reeb graph of f' is loop free. If a diagnostic shows that the domain has no handle, then no loop surgery needs to be done. Otherwise, loop saddles are extracted and the domain is symbolically cut along cutting surfaces. We keep track of the extra pairs of 0-simplices of $\mathcal{R}(f')$ created by each cut. Since the Reeb graph is guaranteed to be loop free, we compute it using a modified

Algorithm 2: Reeb graph algorithm overview.

```
input : Scalar field f : M → ℝ;
begin
    M' ← M, GL = ∅;
    if genus_diagnostic(∂M) > 0 then
        S = find_surgery_loop_saddles(M, ∂M, f);
        foreach s ∈ S do
            M' ← cut(M', cutting_surfaces(s));
            GL = GL ∪ simplex_pairs(s);
    R(f) ← contour_tree(M', f');
    foreach p ∈ GL do
        R(f) ← glue(R(f), p);
end
```

version of the join-split tree algorithm (CSA00). Finally, the loop free Reeb graph is transformed into the correct Reeb graph of the input function by inverse cuts.

Loop surgery

The purpose of *loop surgery* is to symbolically cut the domain such that $\mathcal{R}(f')$ is guaranteed to be loop free.

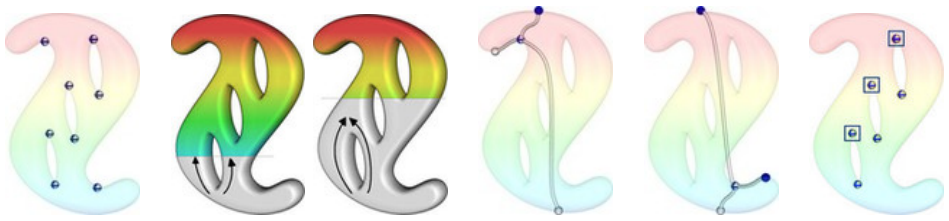
Genus diagnostic We first check if any loop surgery is needed by checking the presence of tunnels on the boundary $\partial\mathcal{M}$. This implements the *genus_diagnostic* procedure of algorithm 2. In particular, we use the Euler formula on $\partial\mathcal{M}$:

$$\chi(\partial\mathcal{M}) = 2 - 2g(\partial\mathcal{M}) = n_v^\partial - n_e^\partial + n_f^\partial \tag{4.8}$$

where n_v^∂, n_e^∂ and n_f^∂ stand for the numbers of vertices, edges and triangles of $\partial\mathcal{M}$. The genus $g(\partial\mathcal{M})$ then gives the number of tunnels in $\partial\mathcal{M}$ and hence the number of cuts needed to ensure that $\mathcal{R}(f')$ is loop free. Loop surgery is needed only if this number $g(\partial\mathcal{M})$ is non-zero.

Loop saddles If the domain is cut at every loop saddle then $\mathcal{R}(f')$ is loop free. In this section, we implement *find_surgery_loop_saddles* of algorithm 2. One technique for finding the loop saddles of f_∂ is computing $\mathcal{R}(f_\partial)$ (using an existing technique such as (PSBM07)), and then identifying each loop of $\mathcal{R}(f_\partial)$ using extended persistence (AEHW04). Alternatively, we found in practice that the benefits of using a simpler technique for finding overall a small superset of surgery loop saddles of f outweigh the cost of performing additional cuts.

Loop saddles of f_∂ are first selected in a three-step process: *(i)* all saddles of f_∂ are identified; *(ii)* we apply property 19 and remove from



(a) $\partial\mathcal{M}, f_\partial$ (b) Sublevel sets of f_∂ (c) Join tree of f_∂ (d) Split tree of f_∂ (e) Loop saddles

Figure 4.13 – *Identifying loop saddles: f_∂ and its saddle points (a). Loops are "absorbed" by sub-level sets of f_∂ (b). Join tree (c) and split tree (d) of f_∂ identify non-loop saddles. Loop saddles are returned (e).*

these the ones that join distinct sub- or sur-level set components; and *(iii)*, we further extract a list of candidate surgery loop saddles of f. The rules we employ resolve degenerate saddles implicitly. These steps are explained in detail below.

(i) Saddles of f_∂ occur at vertices of $\partial\mathcal{M}$. A vertex $x \in \partial\mathcal{M}$ is a saddle if and only if the number of connected components of its lower link is greater than one. This number can be computed by a simple link traversal technique (CMEH*03). The set of all the saddles on the boundary is denoted S_∂.

(ii) We remove saddles S_∂ that do not open or close a loop. To use the result of property 18 in determining whether or not the saddle can be part of a loop, we identify the sub- and sur-level set associated with each connected component of the lower and upper link of a saddle of S_∂. The classification of sub- and sur- level set components is computed by constructing the join tree $\mathcal{J}(f_\partial)$ and the split tree $\mathcal{S}(f_\partial)$. The join sweep that builds the join tree processes vertices of $\partial\mathcal{M}$ in order of increasing function value, maintaining sub-level sets via a Union-Find data structure. In the computed join tree of f_∂, the number of downward 1-simplices of each node is equal to the number of distinct sub-level set components of f_∂ that are joined at the corresponding vertex. Similarly, a split sweep builds the split tree by processing vertices of $\partial\mathcal{M}$ from highest to lowest. The resulting split tree provides the information about the evolution of sur-level set components.

When the number of downward 1-simplices in the join tree equals the number of connected components of the lower link and the number of upward 1-simplices in the split tree equals the number of connected components of the upper link, the saddle does not open or close a loop in $\mathcal{R}(f_\partial)$. These saddles are removed from S_∂.

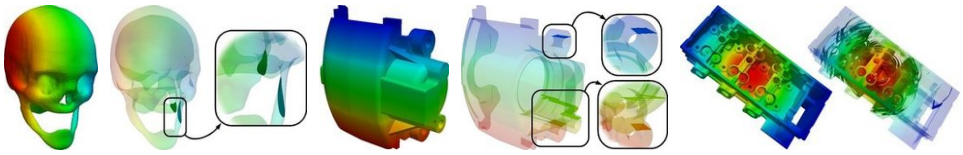


Figure 4.14 – *Examples of cutting surfaces of 3D scalar fields defined on non-simply connected domains: skull (2 handles), brake caliper (3 handles), cylinder head (82 handles).*

(iii) The set S_∂ now exactly contains the loop saddles of f_∂. As described in the previous subsection, each loop $\mathcal{L} \in \mathcal{R}(f)$ maps to a monotone path in $\mathcal{J}(f)$ (or in $\mathcal{S}(f)$) being a subset of the monotone path linking the loop saddles of the corresponding loop $\mathcal{L}_\partial \in \mathcal{R}(f_\partial)$. We identify candidates for *surgery loop saddles* of f by walking upward from the lowest loop saddle of $\mathcal{L}_\partial$ in $\mathcal{J}(f)$ to its highest loop saddle (blue squares in Figure 4.13(e)) and extracting the list of 1-saddles of f that map to the interior of 1-simplices of $\mathcal{J}(f)$ (boundary loop saddles are paired on $\mathcal{J}(f)$ based on their function values, yielding possible over estimations when boundary loops overlap through ψ).

Cutting surfaces We symbolically cut $\mathcal{M}$ through a sequence of symbolic cuts, implementing the procedure *cutting_surfaces* of algorithm 2. According to property 19, to ensure that $\mathcal{R}(f)$ be loop free, every 1-saddle must join distinct sub-level set components. A surgery loop saddle s of f does not have this property, therefore we perform symbolic cuts on $\mathcal{M}$ such that each connected component of the lower link of s have a unique sub-level set component.

A symbolic cut is a contour traversal that updates pointers in the tetrahera that are crossed. A cutting surface S_i is a simple data structure with a unique identifier that is the record of the symbolic cut. Let s_i be a surgery loop saddle with value $f(s_i)$ and let n be the number of connected components of $Lk^-(s_i)$. Cutting surface traversals are started for $(n-1)$ connected components $Lk^-(s_i)$, in particular at a tetrahedron adjacent to each of these. Each symbolic cut produces a new sub-level set component, which is recorded in the cutting surface data structure.

To keep track of the symbolic cuts in the rest of the algorithm, each tetrahedron crossed by any cutting surface stores a pointer for each of its vertices to the highest cutting surface passing below and the lowest cutting surface passing above the vertex. Additionally, each vertex is marked with a *top* flag (Figure 4.15, blue circle) if it lies above a cutting surface crossing the tetrahedron, and also a *bottom* flag (Figure 4.15, red disc) if it lies below a cutting surface crossing the tetrahedron. Finally, each saddle

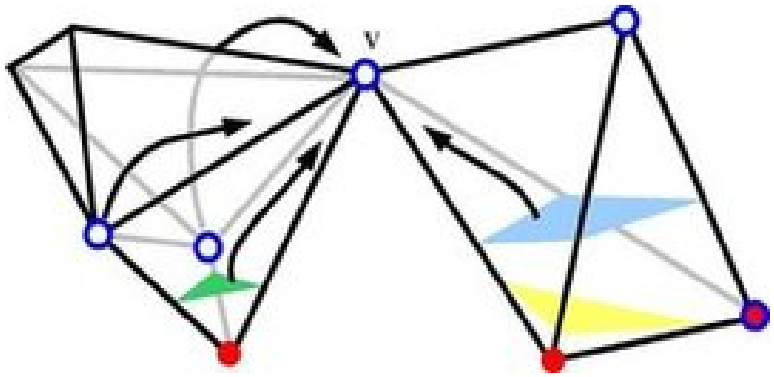


Figure 4.15 – *Simulating symbolic minima in the presence of cutting surfaces (green, blue and yellow) when visiting a vertex v during the join tree construction. Vertices with a* top *(respectively* bottom*) flag are marked with a blue circle (respectively a red disc).*

that generates symbolic cuts stores pointers to the corresponding cutting surfaces.

Loop free Reeb graph computation

By property 19, a loop free Reeb graph can be computed using a contour tree algorithm. Since we cut $\mathcal{M}$ only symbolically, we use a modified version of the join-split algorithm (CSA00) that behaves "as if" $\mathcal{M}$ were actually transformed into $\mathcal{M}'$. Building the loop free Reeb graph using the modified join-split algorithm implements the procedure *contour_tree* of algorithm 2. This modified contour tree algorithm simulates the cuts and uses a Union-Find data structure (using path compression and union by rank) implemented to return the highest element of a set.

Vertices are processed in order of increasing function value, and each vertex v is added to the join tree and a new set is created in the Union-Find. If the star of v contains tetrahedra pointing to cutting surfaces, we simulate $\mathcal{M}$ being cut by simulating the existence of a new sub-level set and a new minimum by adding each cutting surface S_i to the join tree and the Union-Find. The tetrahedra in the star of v that have a vertex in the lower link of v are iterated upon. For each such tetrahedron T, if there is a cut S_i crossing T, we pick the highest S_i that is below v, as illustrated in Figure 4.15 (black arrows emanating from cutting surfaces). This is a constant time operation due to having stored pointers in each tetrahedron in the cutting surface computation. We perform a find and then merge the sets, also adding an arc to the join tree. This simulates having cut $\mathcal{M}$ by S_i, since it disconnects part of the lower link, and instead connects v to an "artificial" minimum, which is the node returned by the Union-Find. Next, the vertices in the lower link of v that were not disconnected by any

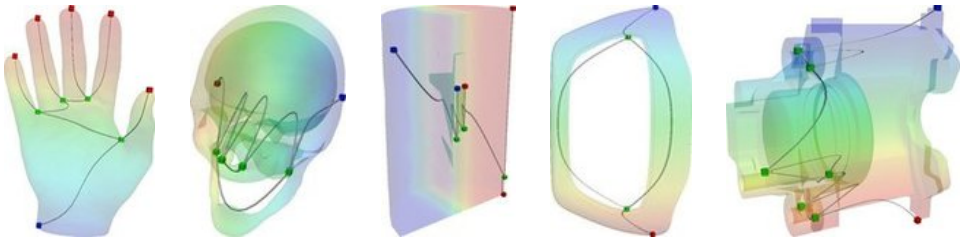


Figure 4.16 – *Examples of Reeb graphs of scalar fields defined on the experiment data-sets. Persistence-based simplification and arc smooth embedding can optionally be computed in a post-process.*

cutting surface are processed (black arrows emanating from vertices in Figure 4.15). If no tetrahedron in the lower star of v is crossed by a cutting surface (*i.e.*, v did not get marked as *top*) then the lower link of v can be processed with no changes to the algorithm in (CSA00). The join tree of f' is returned. The split tree is computed symmetrically, and merging the two trees occurs exactly as in the join-split algorithm. This computes the loop free Reeb graph $\mathcal{R}(f')$.

Inverse cuts

Transforming the loop-free Reeb graph $\mathcal{R}(f')$ into $\mathcal{R}(f)$ requires gluing minimum-maximum pairs of each cutting surface. This implements the procedure *glue* of algorithm 2. For each cutting surface, pointers in the data structure identify the minimum it generated in the join tree and the maximum it generated in the split tree. The two 0-simplices are found in $\mathcal{R}(f')$, and are glued together by concatenating the up-arc of the minimum, and the down-arc of the maximum. This inverts the changes made to $\mathcal{R}(f')$ by loop surgery.

4.2.3 Results and discussion

We implemented Reeb graph computation based on loop surgery in standard C under GNU/Linux. All the experiments presented below were run on a standard desktop computer with a 64-bit 2.83 GHz CPU and 8 GB of memory. Data sets are courtesy of the AIM@SHAPE shape repository and collaborating mechanical design experts.

In our experiments, we compare running times with recent Reeb graph computation techniques for tetrahedral meshes (PSBM07, DN08). We used the original implementations of these approaches, kindly provided by their respective authors. Furthermore, we compared the output of our

approach to that presented in (PSBM07) using the exact same simulation of simplicity (EM90), and found the two algorithms output identical Reeb graphs for all the available datasets.

Time complexity

Let n and N be respectively the number of vertices and simplices of $\mathcal{M}$. Moreover, let n_∂ and N_∂ be respectively the number of vertices and simplices of $\mathcal{M}_\partial$. Finally, let N_S be the number of simplices of $\mathcal{M}$ crossed by cutting surfaces. We present a complexity analysis for each step in our algorithm:

Loop saddle extraction Saddle identification by link traversal requires $O(n_\partial)$ steps. Join tree and split tree computation both require $O(n_\partial log(n_\partial) + N_\partial \alpha(N_\partial))$ where $\alpha()$ is an exponentially decreasing function (inverse of the Ackermann function). Surgery loop saddle candidate extraction takes $O(N)$ steps.

Symbolic cuts computation $O(g(\mathcal{M}_\partial) \times N_S)$ steps: each of the $g(\mathcal{M}_\partial)$ handles of $\mathcal{M}$ generates cutting surfaces and yields a traversal of at most N_S tetrahedra.

Loop-free Reeb graph computation the contour tree algorithm variant requires $O(nlog(n) + N\alpha(N))$ steps (CSA00).

Inverse cuts $O(g(\mathcal{M}_\partial))$ steps: there is an explicit list of cutting surfaces, and gluing the min-max pairs of each takes constant time.

Overall bound $O(nlog(n) + N\alpha(N) + g(\mathcal{M}_\partial) \times N_S)$.

The worst case scenario is reached when both $g(\mathcal{M}_\partial)$ and N_S (loop surgery process) are linear with the size of the mesh, in which case our algorithm will exhibit a strong quadratic behavior. However, with real-life data, $g(\mathcal{M}_\partial)$ is a small constant, resulting in virtually linear scalability in practice.

Performance comparison

We compare the running times of our approach with those of the two fastest previous techniques, presented in (PSBM07) and (DN08). The

Data	Tets	#H	LS	SA (PSBM07)		OS (DN08)	
			Time	Time	Speedup	Time	Speedup
Langley Fighter	70,125	0	0.346 s.	43.70 s.	**126.3**	650.1 s.	**1 879**
Cylinder Head (low res.)	116,274	82	0.660 s.	10.20 s.	**15.45**	257.188s.	**389.68**
Hood (low res.)	120,501	31	0.340 s.	45.77 s.	**134.62**	52.602 s.	**157.71**
Trunk (low res.)	143,366	1	1.866 s.	27.97 s.	**14.99**	406.665 s.	**217.93**
Liquid Oxygen Post	616,050	1	0.686 s.	435.2 s.	**634.4**	15.54 s.	**22.65**
Brake Caliper (med res.)	1,155,317	3	2.242 s.	-	-	180,638 s.	**80,570**
Buckminster Fullerene	1,250,235	0	2.508 s.	9,887 s.	**3,942**	781.0 s.	**311.4**
Plasma	1,310,720	0	2.202 s.	11,983 s.	**5,442**	-	-
S. Fernando Earthquake	2,067,739	0	4.068 s.	15,949 s.	**3,921**	-	-
Brake Disk (high res.)	3,554,828	54	7.798 s.	-	-	-	-

Table 4.2 – *Comparing Reeb graph computation run times. #H stands for the number of handles in the mesh, LS for the loop surgery approach, SA for the streaming approach (PSBM07) and OS for the output sensitive approach (DN08). The loop surgery approach processes in average 428 thousand tets per second and provides an observed average speedup of 6,510. The "-" symbol means that either the process interrupted on a memory exception or that the process had not completed after several days of computation.*

scalar valued data represents a variety of physical phenomena: air turbulence, pressure, liquid oxygen diffusion, rock density, etc. Table 4.2 reports the running times of the three methods on these data.

Our approach achieves significant improvement in terms of running time for each data-set, including those with the highest number of handles, resulting in an average speedup factor of 6,500.

The approaches presented in (PSBM07) and (DN08) rapidly stress and exhibit a quadratic behavior on real-life data. Despite the theoretical quadratic complexity of our loop surgery, the worst-case scenario seems difficult to reach with real-life data.

Asymptotic stress tests

As described previously, the loop surgery procedure has a quadratic worst-case complexity. In practice, for the experiments reported in Table 4.2, we observed this step took 43% of the overall computation in average and that its time requirement was increasing both with the size of the mesh and the number of handles. The pressure field on the trunk data-set is a special case: it is an extremely noisy field that illustrates that we are performing more symbolic cuts than necessary. Overall, we observe that the loop surgery overhead is proportional to the cost of computing the contour tree with real-life data.

Next, we provide a stress test to show the performance of the algo-

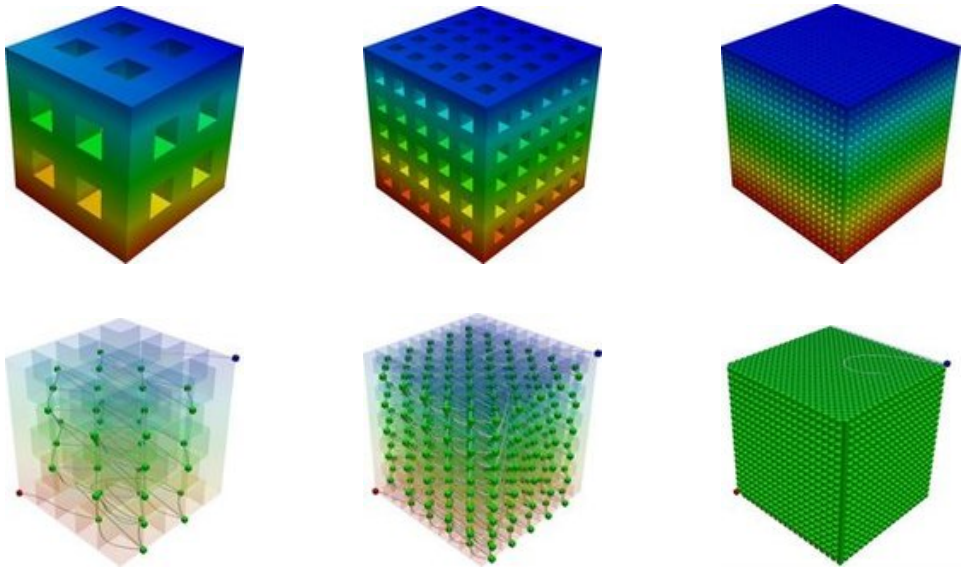


Figure 4.17 – *Handle stress experiment: examples of generated meshes with increasing number of handles (top) and the Reeb graph of their height function.*

rithm in a worst-case scenario. We generate meshes by tetrahedralizing a rectilinear grid on from which rows and columns have been removed, allowing us to increase the number of tets and the number of handles independently. The function value of each vertex is its y coordinate. Figure 4.17 illustrates these meshes. Figure 4.18 shows the running time and the memory footprint as a function of the number of handles and the size of the mesh. This experiment shows that for a constant number of handles, our algorithm scales linearly with the size of the mesh. When the number of handles increases, the linear ratio also increases. The memory footprint follows the same linear behavior as the time execution, due to non-optimized data structures for storing cutting surfaces.

Limitations

A limitation of the algorithm is that it requires the mesh to be manifold with boundary to ensure that the boundary of the mesh is a 2-manifold without boundary. Another is that a given boundary loop can span several 1-saddles that map to the interior of 1-simplices of $\mathcal{J}(f)$. Each of these will be extracted as a surgery loop saddle, yielding potentially several cutting surfaces per handle.

Concluding remarks

In this section, I presented a practical algorithm for fast Reeb graph computation on tetrahedral meshes in $\mathbb{R}^3$. By providing theoretical results on

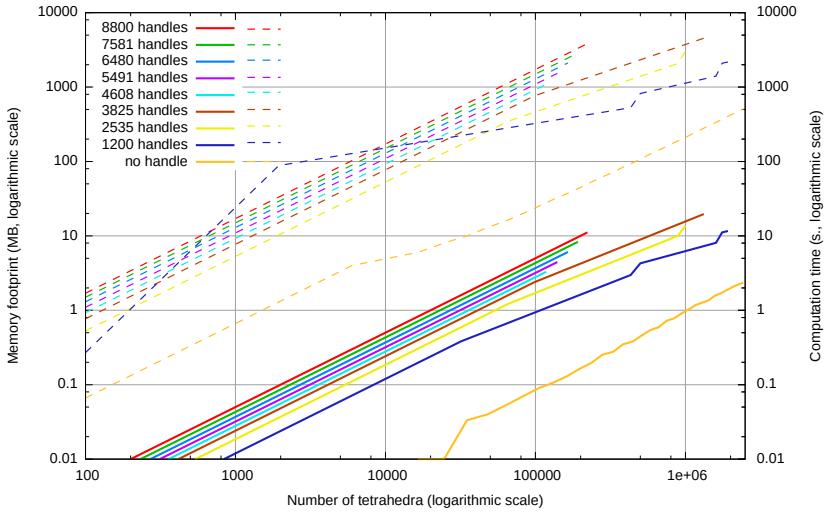


Figure 4.18 – *Running time (solid lines) the memory footprint (dashed lines) when increasing the number of tets for a mesh with constant number of tunnels.*

the topology of such Reeb graphs, we showed their computation could be reduced to a contour tree computation through a technique called loop surgery. Experiments demonstrated in practice the scalability of the algorithm. Moreover, we showed that our approach improves, in terms of running time, the fastest previous techniques for real-life data by several orders of magnitude.

Reducing the computational requirements of Reeb graphs to that of contour trees enables the generalization of the contour-tree based interactive techniques to volumetric meshes of arbitrary topology (as described in the next chapter) and thus opens several avenues for future visualization research. An extension of our approach to volumetric meshes not embeddable in $\mathbb{R}^3$ and of higher dimensions would address a larger class of problems. However, as it is no longer true that such meshes necessarily admit boundary, the loop surgery concept would need to be extended and generalized. Doraiswamy and Natarajan (DN13) later extended this approach to manifolds of arbitrary dimension, by inserting cutting surfaces at each 1-saddle lying in the interior of a 1-simplex of the join tree (and not necessarily those spanned by boundary loops), yielding a more general algorithm at the expense of computing more cutting surfaces than our approach. Due to its fast performances in practice, we considered our loop surgery approach as the reference algorithm for the problem of Reeb graph computation on tetrahedral meshes in $\mathbb{R}^3$ until an optimal time complexity algorithm was introduced three years later (Par12).

Interaction

5

Contents

This chapter describes my contributions for the interactive manipulation of topological abstractions. First, we describe an approach for the interactive simplification of isosurfaces on non-simply connected domains, for visual exploration purposes. This approach is enabled by our fast Reeb graph computation algorithm, described in the previous chapter. Second, we present two algorithms for the editing of topological abstractions in the context of data segmentation. First, I describe how to integrate user constraints in the construction of a discrete gradient to incorporate user knowledge in Morse-Smale complex based segmentations. Second, I describe an approach for the interactive editing of the geometry and topology of a Reeb-graph based segmentation for surface quadrangulation purposes. This chapter presents parts of the results described in the following papers: (TGSP09, STK*09, GGL*14, TDN*12).

As described in Chapter 3, topological abstractions are fundamental data-structures in scientific visualization, to accelerate the computation of geometrical constructions (such as level-sets) or to drive segmentation algorithms. In this chapter, I describe my contributions for the interactive manipulation of topological abstractions. After the data abstraction step (discussed in the previous chapter), interactive manipulation capabilities are often needed in practice to explore the features of interest, prior to their quantitative analysis (discussed in the next chapter). In this chapter, I describe two types of interactive manipulation approaches, for visual exploration and data segmentation purposes.

5.1 Topological simplification of isosurfaces

Scientific data is often affected by noise. When extracting isosurfaces for visual exploration purposes, noise can lead to the appearance of many connected components of isosurfaces, some of which being not relevant application wise but still occluding the components of interest of the isosurface. This occlusion problem can prevent users from visualizing and understanding the geometry of the main features of the data.

Carr et al (CSvdP04) described an approach for the topological simplification of isosurfaces on simply connected domains that addresses this issue. As described in Chapter 3, the contour tree can be used as an indexing data-structure for the fast extraction of level sets. This capability can be nicely combined with persistent homology concepts to only extract the contours corresponding to the most persistent features of the data (Figure 3.19). Carr et al (CSvdP04) described other geometrical measures to prioritize the cancellations of the 1-simplices of the contour tree, such as the volume of their pre-image through ϕ or the integral of f over these pre-images (called hypervolume). This work demonstrated the importance of such measures for segmentation tasks, as they induce more stable and intuitive segmentation parameters.

In this section, thanks to our fast Reeb graph computation algorithm (previous chapter), we generalize this interactive approach to non simply-connected PL 3-manifolds. In particular, we focus on the analysis of pressure fields in mechanical design (where the majority of meshes have handles), a case study where contour-tree based techniques could not previously apply. One experiment in the process of mechanical design involves the analysis of the resistance of mechanical pieces made of different materials to pressure stress. In such experiments, mechanical experts first

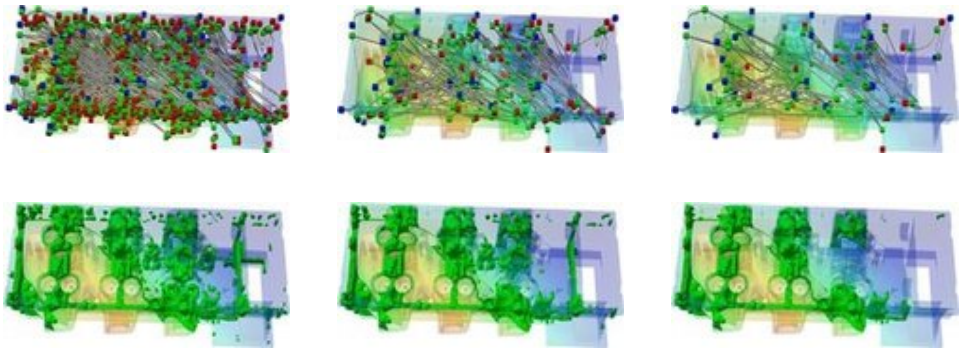


Figure 5.1 – *Topologically clean isosurface extraction on a pressure stress simulation on a cylinder head. The Reeb graph is progressively simplified (from left to right) with increasing hyper-volume scale. As a result, small components (noise) of the considered isosurface are progressively removed (bottom) and the most important features are progressively highlighted (215, 58 and 9 connected components).*

consider simulation previews computed on low resolution meshes. This step illustrates the approximate behavior of the material. Its understanding is crucial to define correct parameters for the actual simulation at high resolution. However, as shown in Figures 5.1 and 5.2, this preview can be noisy, making its interpretation difficult.

We overcome this problem with a fast topologically clean isosurface extraction system. First, the Reeb graph of the low-resolution pressure stress function is computed in a pre-process. Then, local geometric measures (CSvdP04) (extended to tetrahedral meshes) are computed for each arc of the Reeb graph. Then, users may select thresholds for geometric measures to simplify the Reeb graph, as described in (PSBM07). This filtering of the arcs in our examples took at most 0.03 seconds. Our approach maintains degree two nodes in the Reeb graph, representing regular vertices of the function. Consequently, the Reeb graph provides a seed vertex for each contour the user wants to display. We store the non-simplified arcs of the Reeb graph in a balanced interval tree. An isosurface extraction query consists of searching in this tree for a valid seed set. In our examples, this is performed in less than nanoseconds, starting standard isosurface traversal techniques at these seeds in interactive times.

Figures 5.1 and 5.2 illustrate this process on pressure stress functions computed on a brake disk and a cylinder head, where the user progressively increases a simplification threshold with the hyper-volume measure (CSvdP04). The Reeb graphs are simplified, and as a result, the small connected components (noise) of the queried isosurface are progressively removed and the most important features are highlighted. This enables a direct visualization of the major trends of the simulation.

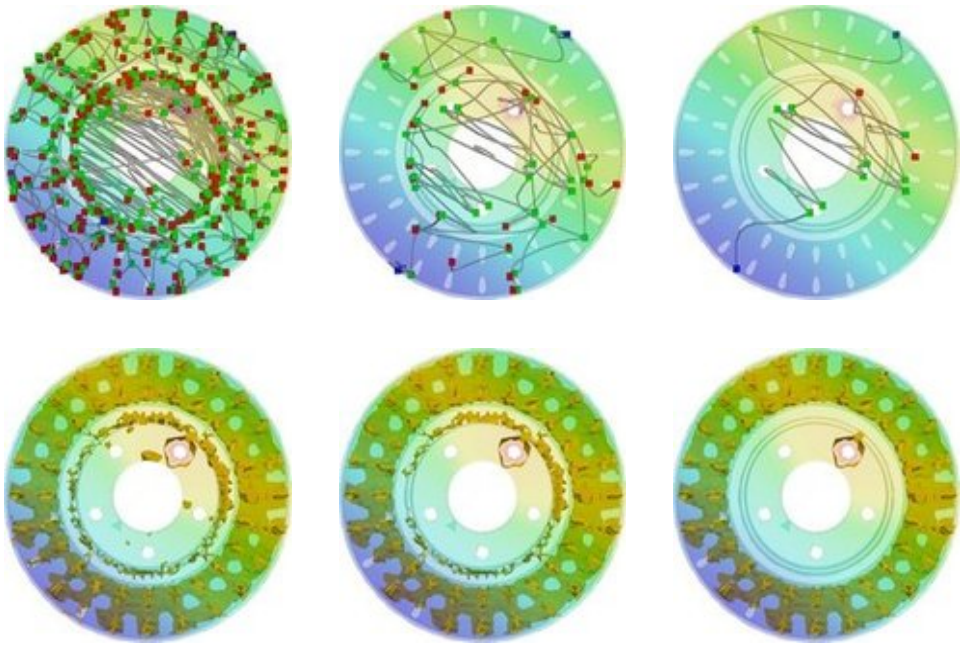


Figure 5.2 – *Topologically clean isosurface extraction on a pressure stress simulation on a brake disc. The Reeb graph is progressively simplified (from left to right) with increasing hyper-volume scale. As a result, small components (noise) of the considered isosurface are progressively removed (bottom) and the most important features are progressively highlighted (92, 8 and 2 connected components).*

Based on this approach, we introduced next an isosurface based visualization widget for a large-scale simulation monitoring system. Large scale numerical simulations running on super-computers are often analyzed online with advanced monitoring systems (KVP*08), providing the end users with real time quantitative indicators describing the behavior of the simulation, allowing them to identify instantly possible run-time issues (mistakes in parameter settings for instance). However, the interpretation of these indicators require strong user expertise and sometimes problematic configurations are difficult to read from those. Therefore, there exists an application need for on-the-fly visualizations of numerical simulations for monitoring purposes. However, large scale simulation time steps are usually too large to be interactively transfered to a remote workstation for visualization. Moreover, due to their size, not all of the time-steps of such a simulation can be stored on the super-computer where they have been generated. This prevents remote rendering and interaction and requires in-situ visualization generation.

We addressed this problem by designing a prototype (STK*09) for the eSimMon monitoring system (KVP*08) developed by Oak Ridge National Laboratory, capable of generating in-situ isosurface renderings (Fig-

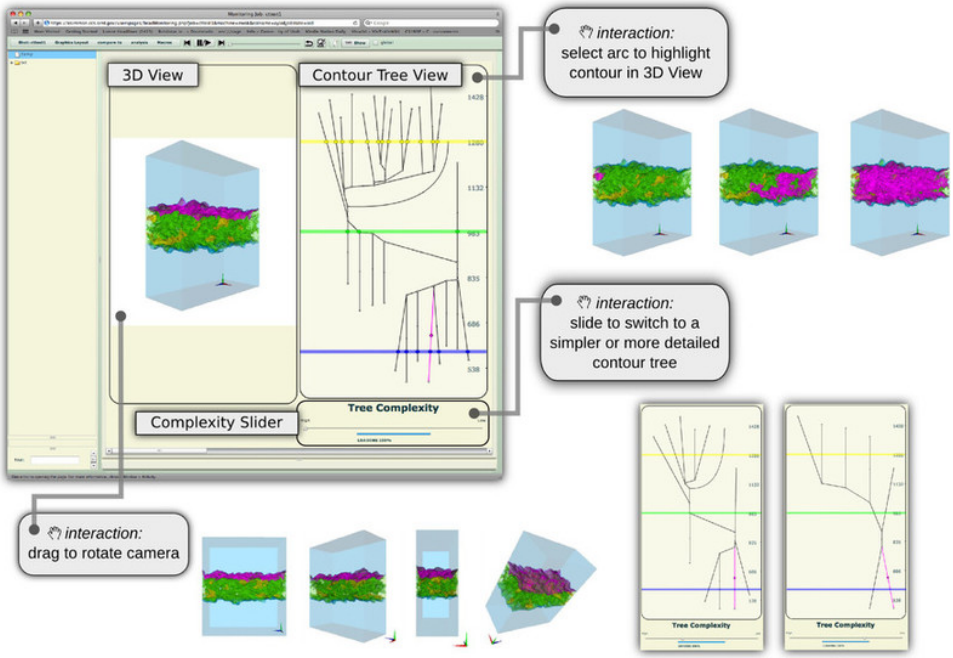


Figure 5.3 – *Isosurface based visualization widget for the eSimMon (KVP*08) web-based simulation monitor: isosurface rendering (top left), Reeb graph planar display (top right), persistence slider (bottom right). Other view examples are provided on the side.*

ure 5.3). The visualization of an isosurface is dictated by a number of parameters such as view point and isovalue and possibly topological simplification threshold. Therefore we implemented an in-situ algorithm that finely samples in batch mode this parameter space (26 view points, 4 level of possible topological simplification, plus the individual extraction of a contour for each 1-simplex of the Reeb graph, in purple in Figure 5.3) and generates for each parameter combination an in-situ offline 2D rendering at a resolution of 1024×1024. Even by finely sampling this parameter space, the data size of the output collection of 2D renderings for a given time-step is configurable and in practice guaranteed to be orders of magnitude smaller than that of the time-step itself. In our experiments, the space required for this collection of 2D renderings was 140 Mb. Next, a web-based client integrated into eSimMon enables the user to explore this collection remotely by emulating the interactive control of these parameters as showcased in Figure 5.3. Note in particular that only the 2D renderings selected by the user are transferred to the web-based client. This technique was enabled by our fast isosurface extraction and simplification algorithms described previously. This prototype enabled simulation users to obtain qualitative visual insights from their running simulations. A similar strategy has been developed independently by Kitware Inc. five years later in its visualization system ParaView Cinema (Kit14).

In our experiments, only data-sets of moderate size were considered to enable the development of this proof-of-concept prototype. As discussed in Chapter 8, the in-situ processing of real-life sized data-sets raises a number of important algorithmic challenges that require to revisit entirely Topological Data Analysis algorithms.

5.2 Interactive editing of topological abstractions

As discussed in Chapter 3, topological abstractions are fundamental data-structures in scientific visualization for data segmentation purposes. In particular, the Reeb graph and its variants are well suited when the boundary of the regions of interest align with level-sets, while the Morse-Smale complex is well suited when features (or their boundaries) align with the gradient. In many applications, such segmentations often directly correspond to meaningful segmentation application-wise, with excellent classification scores. However, such approaches still result in general in the identification of false negatives as compared to a manual labeling by a domain expert. In the following, I describe two approaches for the interactive editing of topological abstractions, in order to integrate users' knowledge in the process.

5.2.1 Morse-Smale complex editing

As described in Chapter 3, when using the Discrete Morse Theory setting, the first step for the computation of the Morse-Smale complex involves the computation of a discrete gradient (Figure 3.20). In this sub-section, I describe the first approach for the computation of a discrete gradient that *conforms* to some alignment constraints provided automatically or interactively by a user. Such a mechanism enables to correct in a pre-process potential false positives in the segmentation, while still benefiting from the multi-scale nature of persistent homology concepts applied to the Morse-Smale complex. To define the concept of a conforming Morse-Smale complex, we first introduce conforming discrete gradient vector fields. Given a regular cell complex K, let $\hat{f} : K^0 \rightarrow \mathbb{R}$ be the input scalar data evaluated on the vertices of K and let $I : K \rightarrow S$ be a surjective map from cells of K to an index set $S \subset \mathbb{Z}$.

Conforming Discrete Gradient We define a discrete gradient vector field V as *conforming to a map* I if and only if for each discrete vector

$\langle \alpha^i, \beta^{i+1} \rangle \in V$, $I(\alpha^i) = I(\beta^{i+1})$, meaning both the head and tail of a discrete vector have the same label in I. We remark that the space of discrete gradient vector fields is the same as the space of conforming discrete gradient vector fields. This is easily seen, since (1) every conforming discrete gradient vector field is also a discrete gradient vector field, and (2) any discrete gradient vector field V is also a conforming discrete vector field under the map $I(\alpha) = 1$. Since the Morse-Smale complex $\mathcal{MS}(f)$ is uniquely determined by a discrete gradient vector field V, we say $\mathcal{MS}(f)$ *conforms* to a map I if and only if V also conforms I.

Algorithm

Computing discrete gradient vector fields so that they conform to any arbitrary map I involves only a slight variation of previous techniques. In particular, the added restriction that a gradient vector $\langle \alpha, \beta \rangle$ can only be created if $I(\alpha) = I(\beta)$ by definition creates a conforming discrete gradient. Note that any valid discrete gradient V can be converted to a conforming one by simply making each cell α and β critical when $\langle \alpha, \beta \rangle$ and $I(\alpha) \neq I(\beta)$. However, such a trivial modification does not represent how the flow behaves when restricted to the boundary of a labeled section. In the following algorithm, we create a discrete gradient vector field that conforms to I but nevertheless avoids creating critical cells where a discrete gradient vector $\langle \alpha, \beta \rangle$ can be created where $f(\beta) \leq f(\alpha)$ and $I(\alpha) = I(\beta)$, i.e., there exists *some* gradient flow restricted to the label $I(\alpha)$.

Our algorithm is inspired by Robins et al.: first create a discrete gradient vector for a vertex-edge pair and then perform simple homotopic expansions in the lower star (RWS11). The main difference is that we only consider for pairing those cells that also share the same label. Robins' algorithm has a strict ordering for performing homotopic expansions in the lower star, and straightforward restriction of pairing based on labels would create many extra critical cells. We instead take a very pragmatic approach to reducing the number of spurious critical cells produced by reordering the homotopy expansion to perform any possible pairing in the lower star that is a simple homotopy expansion (the inverse of homotopic collapses (Coh73)) yet also conforms to the labeling.

In the following, we denote a cell that has been identified as critical by pairing it with itself, for example, $\langle \alpha, \alpha \rangle$. Furthermore, a cell is *assigned* if and only if it has been identified as critical or paired in a discrete gradient vector. The function $\#UCF(\gamma)$, (Number of Unassigned Conforming

Faces) counts the number of faces $\beta \leq \gamma$ of a cell γ restricted to the lower star $St^-(\alpha)$ such that β has not been assigned and $I(\beta) = I(\gamma)$.

Algorithm 3: Conforming gradient construction algorithm.

input : Scalar field $\hat{f} : K \to \mathbb{R}$, map $I : K \to \mathbb{Z}$;
output: Conforming discrete gradient V;

begin
  $V \leftarrow \emptyset$;
  foreach $\alpha \in K^0$ **do**
    $S \leftarrow \{\beta^1 \in St^-(\alpha) \mid I(\alpha) = I(\beta^1),\ \alpha \leq \beta^1\}$;
    if $S = \emptyset$ **then**
      $V \leftarrow V \cup \langle \alpha, \alpha \rangle$;
    else
      $V \leftarrow V \cup \langle \alpha, \beta^1 \rangle$, $\beta^1 \in S$ is in direction of steepest descent;
    foreach $i \in [1, \ldots, d]$ **do**
      while $\exists$ *unassigned* $\beta^i \in St^-(\alpha)$ **do**
        while $S^{i+1} \leftarrow \{\gamma^{i+1} \in St^-(\alpha) \mid \#UCF(\gamma) = 1\} \neq \emptyset$ **do**
          $V \leftarrow V \cup \langle \beta^i, \gamma^{i+1} \rangle$, β^i is the assigned conforming face of $\gamma^{i+1} \in S^{i+1}$;
        if $\exists$ *unassigned* $\beta^i \in St^-(\alpha)$ **then**
          $V \leftarrow V \cup \langle \beta^i, \beta^i \rangle$, β^i is unassigned;
end

The algorithm processes each vertex independently, first creating a vertex-edge vector in the direction of steepest descent, restricted to the set of edges sharing the same segmentation label as the vertex. If no pairing for the vertex is possible, it is made critical.

Next, simple homotopy type expansions are performed in order of increasing dimension (where d stands for the dimension of the domain), again restricting possible candidates for pairing to those sharing the same segmentation label. For each dimension i, while there exists unassigned i-cells in the lower star of α, simple homotopy expansions of an unassigned i-cell with unassigned $i+1$ cells are attempted, marking the i-cell critical when such an expansion is not possible. The test to check if there exist unassigned i-cells in the lower star of α can be implemented by placing the i-cells in $St^-(\alpha)$ in a list, whose size is typically bounded by a small constant. The output is guaranteed to produce a discrete gradient vector field, since all pairings are restricted to the lower star of a vertex and a homotopy expansion is only performed when all faces of the i-cell have previously been assigned, and the $i+1$-cell has only one unassigned face. These two conditions along with the fact that every cell of the domain is either paired or marked critical ensure that all V-paths produced are monotonically decreasing and V is acyclic, hence a discrete gradient vector field (just as in (RWS11)). Algorithm 3 can be applied to every vertex in the domain in an embarrassingly parallel manner.

This same kind of modification is also possible for discrete gradient

construction algorithms other than Robbin's (GBHP08, GBP12). Overall, just as in our *ConformingGradient*() algorithm, the modifications to these include checking the segmentation label when assigning the first vertex-edge gradient vector, and subsequently checking the label when assigning the higher dimensional vectors during simple homotopy expansion. Furthermore, slight reorderings may be necessary to remove spurious critical points. Once the discrete gradient vector field has been computed, any one of the algorithms for traversal of the discrete gradient field (GBHP08, RWS11, SN12) can be used to compute the conforming Morse-Smale complex.

Editable Morse-Smale complexes

The *ConformingGradient*() algorithm for computing conforming Morse-Smale complexes can be used to enable editing of Morse-Smale complexes. Editing can be used by domain experts to correct errors when doing feature extraction. For instance, in bio-medical imaging, overlapping morphological structures, insufficient dye penetration, shadows, and light diffraction often contribute to poor feature representation during analysis. In this case, the image data contains insufficient or incorrect information for an accurate Morse-Smale complex based feature identification. However, a domain scientist may understand the deficiency and have the interpretation skills necessary to understand what the segmentation *should* be. Often, long pipelines of image processing filters are used to prepare image data for semi-automated segmentation, such as deconvolution, noise removal, color/contrast correction, thresholding, and smoothing. Furthermore, the Morse-Smale complex itself may be simplified and filtered to extract the desired features. Each stage of this pipeline may depend on several parameters, each one affecting the accuracy and precision of the resulting feature identification, when compared to the gold standard of a domain expert manually identifying features. While most features can be extracted with such an approach, there may still remain problematic cases. In such an instance, we allow the user to edit the segmentation iteratively as part of the analysis pipeline.

Once a Morse-Smale complex has been computed on a d-dimensional domain, its ascending and descending d-manifolds define an origin/destination map. Then, reconstructing the Morse-Smale complex using that map as a constraint with our conforming gradient algorithm results in the same Morse-Smale complex. However, it is also possible

to *edit* the maps prior to the second computation, effectively editing the reconstructed Morse-Smale complex. The combinatorial nature of the algorithm guarantees that the result is a valid Morse-Smale complex that conforms to the new edits. In this following, I restrict the discussion to two-dimensional images.

Identity map Let V be a discrete gradient vector field computed with any algorithm on the mesh K and function $\hat{f}$. There are many maps I such that *ConformingGradient*() produces the exact same discrete gradient vector field V. Two trivial examples are $I(\alpha) = 1$ for any cell α, and the map taking each distinct gradient vector and assigning both cells a their own label. Such maps are called *identity* maps. Our approach for editing the Morse-Smale complex is to generate an identity-like map from the Morse-Smale complex as described below, allow the user to modify that map, and then recompute the Morse-Smale complex with the conforming algorithm.

Termination map We construct the initial map based on the origin/destination of V-paths in V. In particular, the ascending and descending manifolds of $\mathcal{MS}(f)$ allow us to construct a *termination map*, $\omega : K^{0,d} \rightarrow S \subset \mathbb{Z}$, that stores a labeling of the destination of each vertex and d-cell. Let $M_n = \{\alpha_0, \alpha_1, ..., \alpha_k\}$ be the set of critical vertices, i.e., the minima. For each minimum α_j, we compute the set of vertices belonging to its ascending d-manifold, and assign those the label j. Symmetrically, we compute the descending manifolds of critical d-cells, i.e. the maxima, and assign all d-cells the same label. Given a Morse-Smale complex $\mathcal{MS}(f)$ constructed from a discrete gradient field V, the termination map constructed from its ascending/descending manifolds as described above is denoted ω_f. Note that ω_f can be seen a the vertex-based (respectively face-based) segmentation of the domain induced by the Morse complex of $-f$ (respectively by the Morse complex of f).

Boundary map We use ω_f to identify those cells in K that are on the boundaries of ascending/descending d-manifolds. We can extend ω_f to construct a new label map ∂_f over all simplices that encodes the appropriate boundary information. In this case, $\partial_f : K \rightarrow \{0, 1, 2, 3\}$ and simply records whether a cell is the boundary of ascending d-manifolds, descending d-manifolds, both, or neither (as illustrated in Figure 5.4):

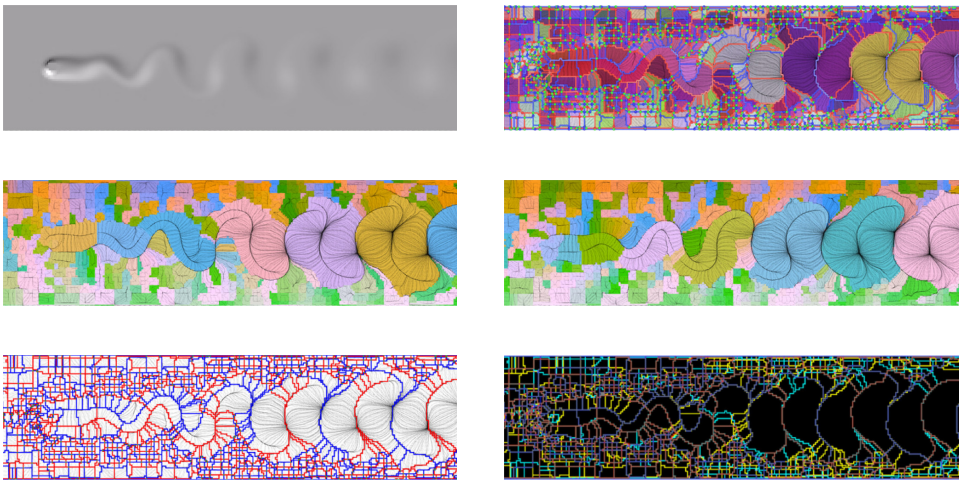


Figure 5.4 – *Example of termination maps and boundary map for the vorticity field of the 2D von Kármán street (top left). The Morse-Smale complex of the data is first computed (top right) to compute the minima (middle, left) and maxima (middle right) termination maps. The separatrices of the Morse-Smale complex (bottom left) are used to construct the boundary map (bottom right).*

$$\partial_f(\alpha) = \begin{cases} 0 & \text{if } \alpha \text{ is not a boundary cell} \\ 1 & \text{if } \alpha \text{ is a boundary between ascending } d\text{-manifolds} \\ 2 & \text{if } \alpha \text{ is a boundary between descending } d\text{-manifolds} \\ 3 & \text{if } \alpha \text{ is a boundary between both} \end{cases}$$

User edits Given the initial termination map ω_f, we allow the user to modify it directly into ω'_f. After each edit, the boundary map ∂'_f is recomputed. Then, the Morse-Smale complex $\mathcal{MS}(f')$ is computed from the gradient field produced by *ConformingGradient*(). If persistence simplification/filtering was used to generate $\mathcal{MS}(f)$, that same level of simplification is performed on $\mathcal{MS}(f')$. Practically, the difference between $\mathcal{MS}(f)$ and $\mathcal{MS}(f')$ is restricted to the region of ∂'_f modified by the user. Note that user edits may force changes to the Morse-Smale complex in non-trivial ways, such as splitting regions and changing the connectivity. For edits such as moving the boundaries of regions, critical points are necessarily added when there is no monotonic path that can be found on the boundary.

An additional simplification rule must be observed to ensure that the edits are not immediately removed as low-persistence features: a critical point of $\mathcal{MS}(f')$ that is a boundary of ω'_f but not of ω_f can only be canceled with another similarly labeled critical point. Figure 5.5 illustrates

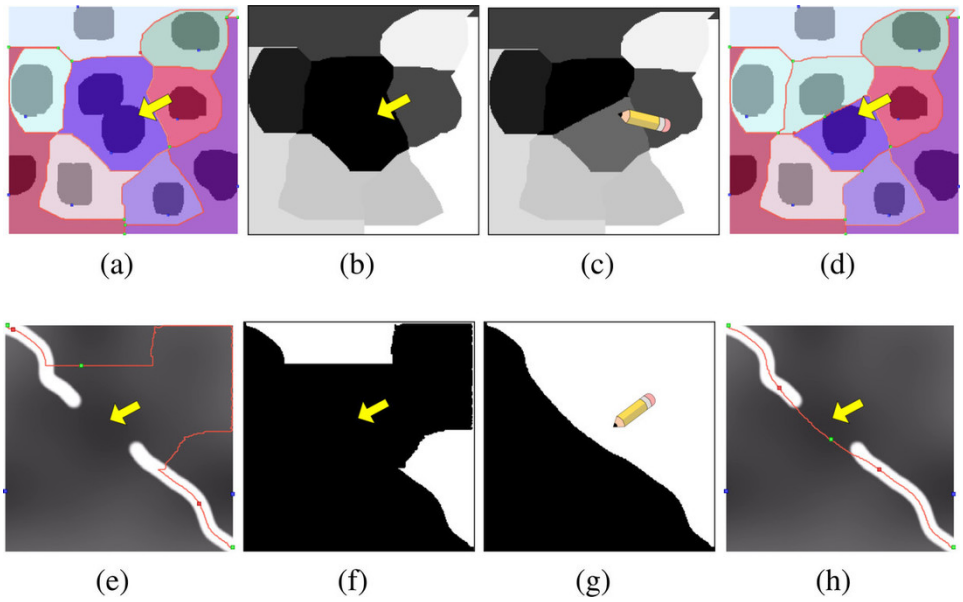


Figure 5.5 – *The Morse-Smale complex is computed for a simple example showing dark blobs on a light background (a). Two overlapping features are identified as part of the same basin. This Morse-Smale complex generates a termination map (b), that can be edited by a user (c) to split this region. The edited map is used to generate a new MS complex (d) with the false negative corrected. Similarly, a light ridge-like structure (e) is disconnected, resulting in a ridge reconstruction with poor geometric embedding. The termination map (f) is again edited (g) to reconstruct the desired embedding (h).*

this process for two simple examples. In the first (a-d), overlapping features cause a false negative to appear, where the finest resolution Morse-Smale complex image insufficiently segments the domain. The user modifies the termination map ω_f, and the complex is regenerated with the false negative corrected. In the next example (e-h), the geometric embedding of an arc of the Morse-Smale complex is fixed by modifying the spatial extents of a label in ω'_f.

A Histological Example Applying Morse-Smale complexes for the automated segmentation of nuclei in phenotypic analysis of histological sections is an active area of research (SCD13). We apply a straightforward approach, identifying nuclei as simplified basins in a processed image. However, the basins identified often may not sufficiently segment the domain to separate all the nuclei. Our goal is to remove all false negatives using a semi-automated approach. As Figure 5.6 illustrates, we apply grayscale conversion, thresholding, and smoothing to the initial image data, and compute basins of the Morse-Smale complex simplified to some persistence threshold.

This pipeline results in some false negatives where two or more nuclei

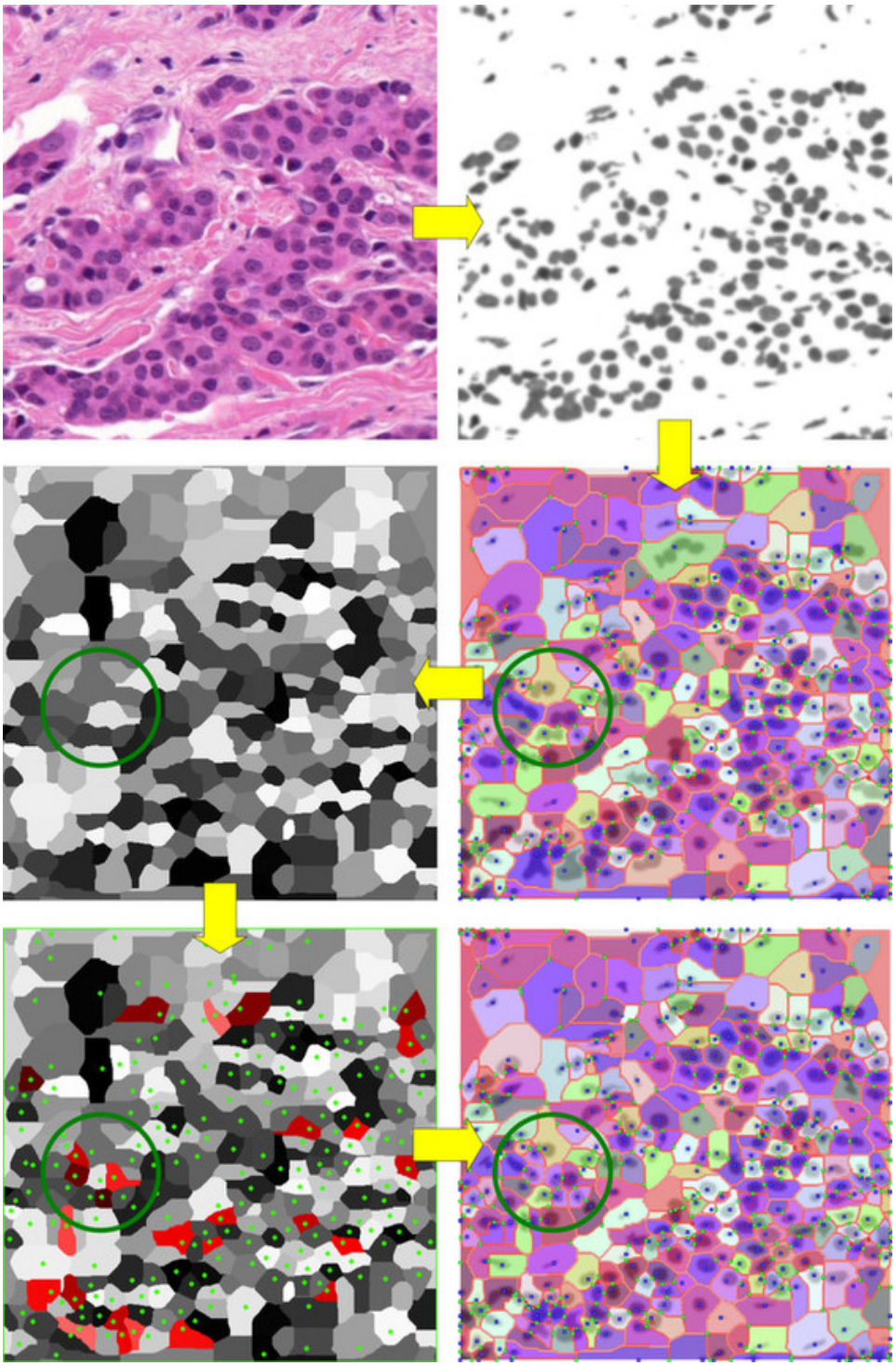


Figure 5.6 – *A section of tissue is imaged in a histology study. We apply grayscale conversion, thresholding, and smoothing to generate an input for an initial MS complex computation. The termination map is edited to match a domain expert's labeling (red regions, bottom left), and a complex is regenerated that has no false negatives. The green circle highlights a region that is corrected through editing.*

overlap, as compared to a labeling by a domain expert (WHS*12). The computed Morse-Smale complex segmentation is used to generate a termination map ω_f, which is edited to match the input of a domain expert. The boundary ∂'_f is computed from ω'_f, and finally the complex $\mathcal{MS}(f')$ is recomputed and simplified, with the new features present. While the Morse-Smale complex that results contains no false negatives, it still contains false positives. However, as it is still a valid Morse-Smale complex, further filtering/simplification is possible to account for such errors. Having every nuclei in its own basin makes it later possible to identify which basins contain nuclei and which do not, for instance, by applying standard filtering/simplification pipelines.

Concluding remarks

This sub-section introduced an algorithm to construct Morse-Smale complexes that conform with a user-supplied segmentation. An investigation of the breadth of interactions that must be supported and user interface requirements to make such an approach a viable component of the analyst's toolbox is future work. Additionally, an interesting direction for future work is to restrict the (re-)computation necessary for a local region of influence around the modified segmentation, to enable instantaneous updates of the Morse-Smale complex. Furthermore, editing may introduce non-local effects both in the feature generation and simplification process, and fully exploring these in a robust manner would be a valuable future contribution.

Finally, another application area for future work involves robust mesh generation. The Morse-Smale complex has started to emerge as vehicle for surface meshing (DBG*06, HZM*08). We view conforming Morse-Smale complexes as a first step towards utilizing Morse-Smale complexes for volumetric mesh generation. User edits with a rich segmentation interface could drive the repair of mesh artifacts interactively, similar to the approach presented in the following sub-section.

5.2.2 Reeb graph editing

In Computer Graphics, surface quadrangulation representations are often preferred over triangulations for tasks such as texture mapping or animation. Quadrangulations can be obtained by partitioning the surface in a set of quadrangular charts (which can be further refined as desired). However, end-users need to control the overall layout and topology of this

partitioning (chart number and boundary alignment, position, valence and number of extraordinary vertices) as it can affect the output animation.

In this subsection, I describe an approach for the semi-automatic segmentation of surfaces into quadrangular charts, based on editing mechanisms of the Reeb graph. Given an initial segmentation automatically generated from the pre-image of $\mathcal{R}(f)$ through ϕ, we introduce atomic operations for the control of the number of charts, the geometrical control of their boundaries, as well as the robust control of the number and valence of chart corners. Each of these operations is accompanied with an intuitive user interface and our core algorithms perform edits at interactive rates, enabling users to constructively enhance the quadrangular segmentation of the surface for quad-meshing purposes (further described in the next chapter).

Harmonic scalar fields

In order to control the geometry of the boundaries of the quadrangular charts, we will consider as an input scalar field the solution to an optimization process, integrating user constraints for the definition of the geometry of the level sets. Moreover, since it is often desirable in practice to construct quadrangular charts with smooth boundaries, we will add a regularization term in the optimization.

The solution to the Laplace equation under Dirichlet boundary conditions is a good candidate for such requirements, since it smoothly interpolates sparse constraints on the entire domain. Given a finite set of extrema constraints C along with corresponding target values, this equation is defined as follows:

$$f(c_i) = f_{c_i} \qquad \forall c_i \in C \tag{5.1}$$

$$\Delta f(v) = 0 \qquad \forall v \notin C \tag{5.2}$$

where Δ stands for a discretization of the Laplace-Beltrami operator on surfaces. The scalar fields being solutions of this equation are called *harmonic scalar fields*.

In practice, to compute such harmonic scalar fields, the Laplace-Beltrami operator is discretized using cotangent weights (PP93), which leads to a symmetric positive-definite sparse matrix $L = W - D$ whose elements w_{ij} of W are defined as:

$$w_{ij} = \begin{cases} -\frac{1}{2}(\cot \alpha_{ij} + \cot \beta_{ij}) & \text{if edge } [i,j] \in \mathcal{M} \\ 0 & \text{otherwise} \end{cases} \tag{5.3}$$

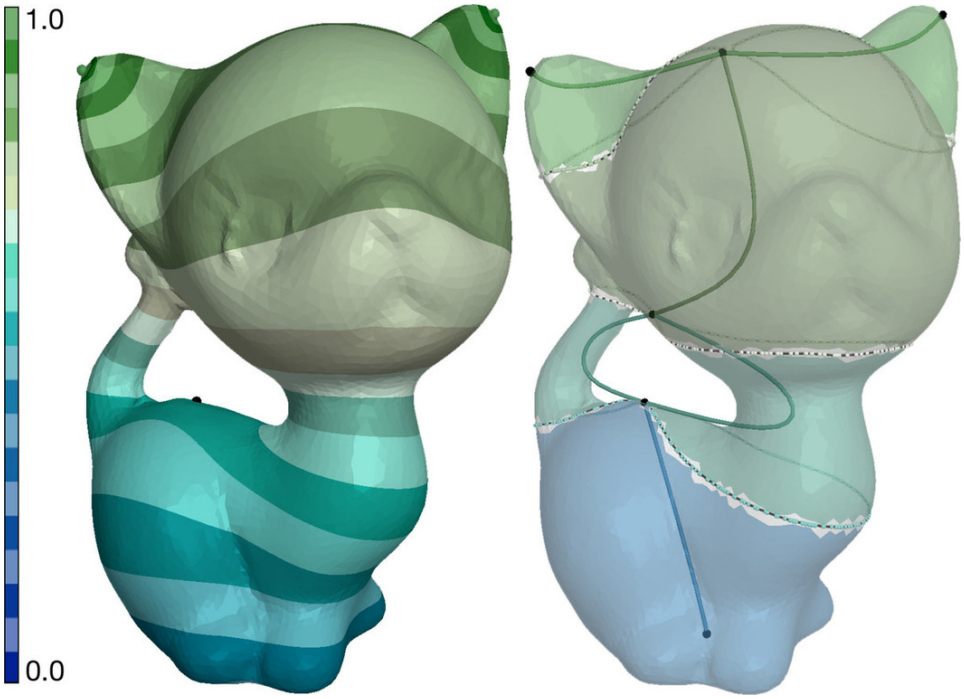


Figure 5.7 – *Harmonic scalar field (left, extrema constraints are located on the ears, top, and at the bottom) and its Reeb graph (right) as well as its pre-image through ϕ (transparent chart colors, right).*

where α_{ij} and β_{ij} are opposite angles to edge e_{ij} and D is a diagonal matrix with elements d_{ii} given by row sums of W.

We make use of the penalty method to impose constraints to the linear system derived from equation (5.2). Consider C, the set of indices of constrained vertices, then the harmonic scalar field is obtained by solving the linear system,

$$(L + P)f = Pb, \tag{5.4}$$

where P is a diagonal matrix with non-zero entries $p_{ii} = \alpha$ only if $i \in C$ and α is the penalty weight ($\alpha = 10^8$ (XZCOX09)). Constrained values are set within the vector b,

$$b_i = \begin{cases} f_{c_i}, & \forall c_i \in C \\ 0, & \forall v \notin C \end{cases} \tag{5.5}$$

where f_{c_i} is the desired scalar value assigned to vertex c_i. The main advantage of using the penalty method to impose constraints is that supernodal schemes (DH09) can be used to update (and downdate) the Cholesky fatorization, making it possible to include and remove constraints efficiently (XZCOX09), at interactive rates. Figure 5.7 shows an example of harmonic scalar field along with its corresponding Reeb graph.

In practice, the set of extrema constraints can be either provided by the

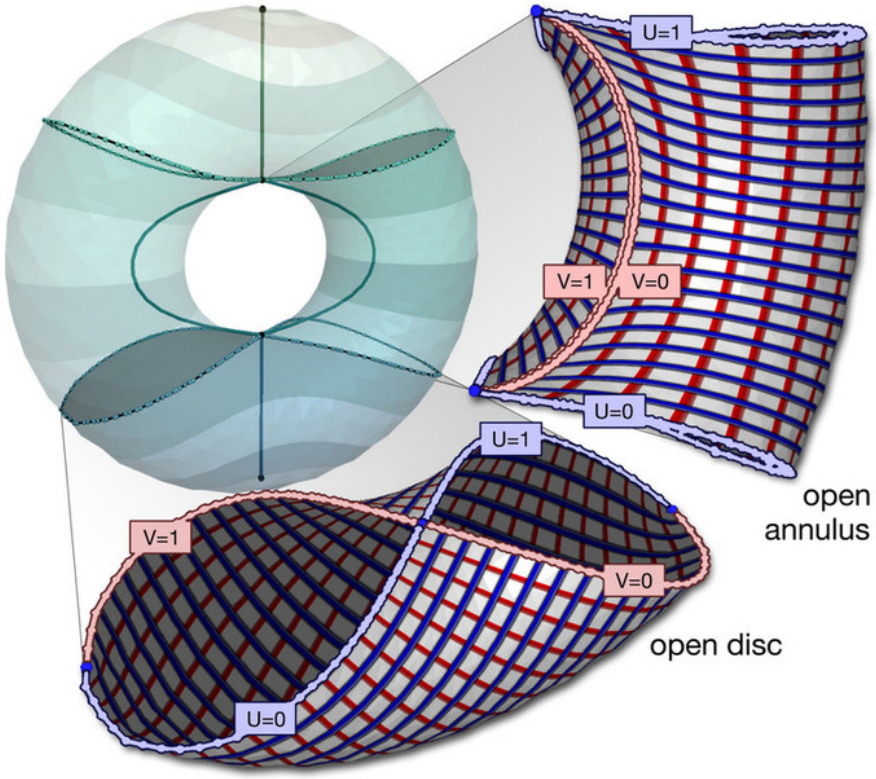


Figure 5.8 – *Our parameterization strategy maps the boundaries of the Reeb charts to the unit square by defining UV Dirichlet boundary conditions within the Laplace system. Open annuli are cut into quadrangular charts by an integral line of f.*

user, or selected as extrema of relevant functions computed automatically, such as the integral of the geodesic distance function (HSKK01).

Reeb-graph based surface segmentation and parameterization

Given a closed PL 2-manifold $\mathcal{M}$ embedded in $\mathbb{R}^3$, a *Reeb chart* is the pre-image by ϕ of the interior of a 1-simplex of $\mathcal{R}(f)$. By construction, Reeb charts are continuous pilings of closed 1-dimensional contours. Since they are the pre-image of the *interior* of 1-simplices, Reeb charts do not include critical contours and are thus open sets with the topology of an open annulus (a connected genus zero surface, with two boundary components excluded). Note that a boundary component collapses to a point if a 1-simplex is linked to the image through ϕ of an extremum. Because Reeb charts are constructed from the regular contours of f, their definition does not require f to be strictly PL Morse (i.e. degenerate saddles are allowed).

Given the segmentation of the surface into Reeb charts, the *Reeb atlas* is defined as the union of the charts with respective local parameterizations. Because Reeb charts have a controlled topology, they are robustly,

Operation	Add Chart	Del Chart	Move Bound.	Add Bound.	Saddle Align.	Frac. Poles	Frac. Saddles
Figure	Fig. 5.10 (top)	Fig. 5.10 (top)	Fig. 5.10 (mid.)	Fig. 5.10 (bot.)	Fig. 5.11	Fig. 5.12	Fig. 5.13
Interaction	Click	Click	Drag	Click	Clicks	Clicks	Clicks + Drag

Table 5.1 – *Summary list of the Reeb atlas editing operations.*

easily and efficiently parameterized with a generic strategy, which will be instrumental for quadrangulation purposes.

Each Reeb chart $\mathcal{M}_i$ of $\mathcal{M}$ is built by duplicating the triangles of $\mathcal{M}$ that fully map to the *interior* of a 1-simplex σ_i via ϕ. *Boundary triangles*, intersected by the critical contours whose image by ϕ is a face of σ_i, are also inserted into $\mathcal{M}_i$, illustrated as grey triangles in Figure 5.7 (right). In order to obtain smooth boundary components for the charts, the *boundary triangles* are shrunk such that their vertices being outside of the chart get snapped along the boundary critical contour (by sliding them along an incoming edge crossing the contour).

A parameterization maps the open annulus $\mathcal{M}_i$ to the unit square by solving two harmonic functions with Dirichlet boundary conditions (Fig. 5.8) using the solver presented previously. The field $U : \mathcal{M}_i \to [0,1]$ is computed to align with the level lines of f by constraining the boundary vertices of $\mathcal{M}_i$, projected to the two critical contours, to either $U = 0$ or $U = 1$ (Fig. 5.8). The orthogonal field $V : \mathcal{M}_i \to [0,1]$ is computed by tracing a cutting integral line along the mesh edges of $\mathcal{M}_i$ guided by the gradient of U, turning the annulus into a quadrangular chart with disc topology. The vertices of the cutting edges are duplicated and assigned values, $V = 0$ and $V = 1$, to map the boundary of $\mathcal{M}_i$ to the unit square.

Each Reeb chart $\mathcal{M}_i$ mapping through ϕ to a 1-simplex σ_i of $\mathcal{R}(f)$ having as face the image by ϕ of an extremum of f (Fig. 5.8) are parameterized differently. The boundary triangles adjacent to the extremum are included within $\mathcal{M}_i$ so that $\mathcal{M}_i$ has a single boundary component and is homeomorphic to a disc. The boundary vertices are segmented into four contiguous polylines and assigned values mapping the boundary to the unit square.

At this stage, the Reeb atlas represents a coarse quadrangular segmentation of the surface. Moreover, each Reeb chart is equipped with its own local parameterization to the unit square, enabling further quadrangular subdivisions. Note that saddles of f can be seen as extraordinary vertices in this segmentation (i.e. vertices whose valence is different from 4).

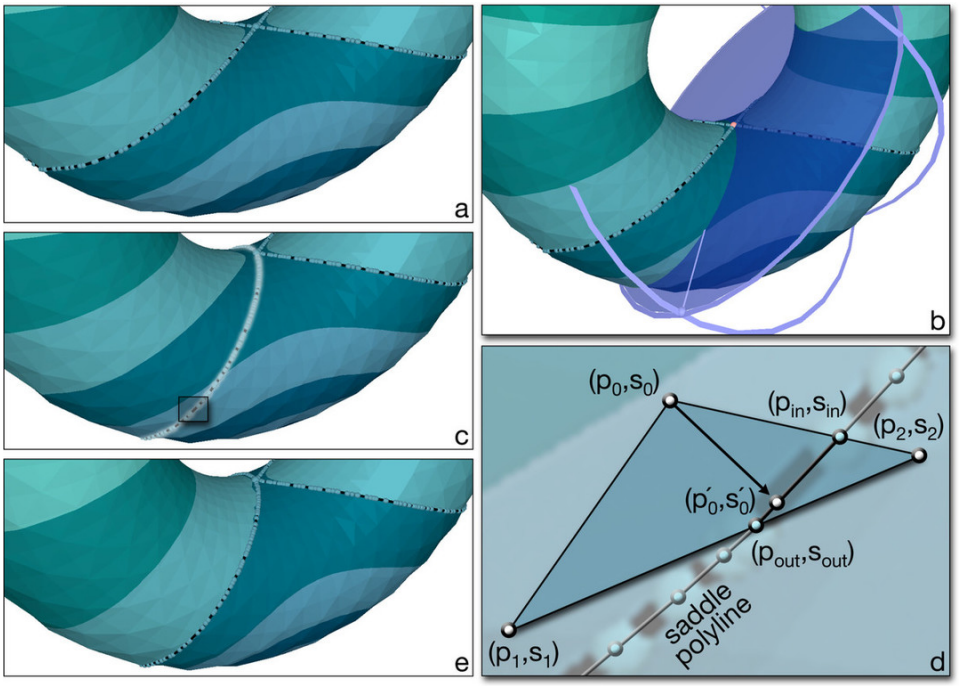


Figure 5.9 – *Editing a chart boundary by saddle contour manipulation: the original level-set curve (a) is modified through the critical contour widget (b) where the mesh-plane intersection describes the new contour geometry (c). Saddle triangles are constrained (d) to ensure the scalar field respects the new critical contours (e)*

Editing operations

In the following, I describe a set of atomic operations (detailed in table 5.1) on the Reeb graphs for the interactive control of the initial quadrangular segmentation provided by the Reeb atlas.

Chart boundary The Reeb chart boundaries are defined by critical contours of f. While the relocation of minima and maxima is well understood, consisting of removing the original constraint and replacing it with a new one at a different location, moving saddle contours requires a bit more machinery but it is however mandatory to edit the geometry of charts' boundaries.

For each saddle contour, additional constraints are added to the Laplace system at the vertices of *saddle triangles* (triangles intersected by the critical contour) to ensure that the scalar field level-sets respect the user designed geometry. Assume that the user modified the geometry of a saddle contour with a scalar value s_c so as to intersect the triangle $t = \{p_0, p_1, p_2\}$ on edges $e_0 = \overline{p_0 p_1}$ and $e_2 = \overline{p_2 p_0}$ (Fig. 5.9). The intersection points $p_{in} = p_0 + \alpha_0(p_1 - p_0)$ and $p_{out} = p_2 + \alpha_1(p_0 - p_2)$ and scalar values $s_{in} = s_0 + \alpha_0(s_1 - s_0)$ and $s_{out} = s_2 + \alpha_1(s_0 - s_2)$, where s_0, s_1, s_2

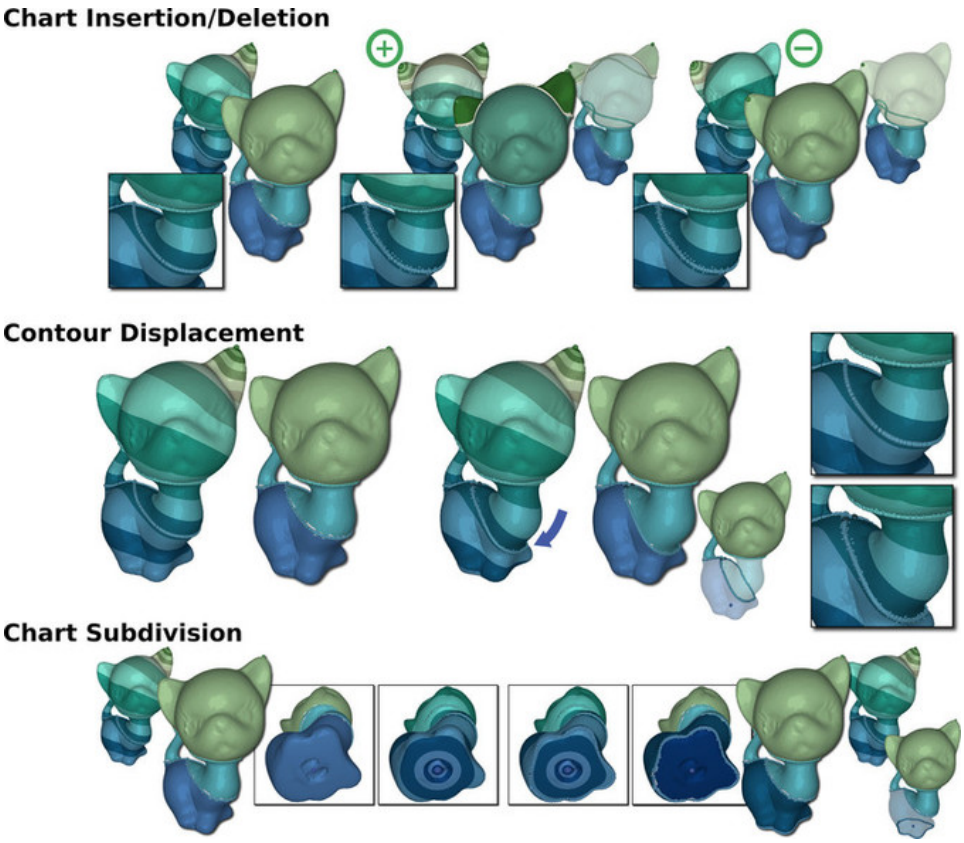


Figure 5.10 – *Global impact of Reeb atlas editing operations. Reeb charts whose triangle list changes after the operation are transparent. Top: chart insertion and deletion (through insertion and deletion of extrema). Middle: contour displacement. Bottom: chart subdivision.*

are associated with the vertices of t, assist in the definition of vertex constraints. The vertex p_i of t is projected onto the segment $\overline{p_{in}p_{out}}$ yielding the point p'_i with a scalar value s'_i. The scalar constraint assigned to p_i for t is $\tilde{s}_i = s_c + (s_i - s'_i)$. The final constraint of each vertex is averaged with values of adjacent saddle triangles. This novel constraint computation enables strict control of the contour of f, aligning to the user's designed polyline.

Note that initially, and also after each editing operation, *all* the saddle contours of f are constrained using the above scheme (even if the contours are not displaced). Then, the effects of the editing operations are localized to the charts of interest. For instance, when displacing a saddle contour at the boundary of a chart of interest, the other boundaries remain in position and conserve their f values. Then to guarantee that the saddle contour displacement does not alter the topology of f, our interface discards any interaction that generates a contour which is not a closed loop, or that makes the contour overlap another saddle contour or sweep an

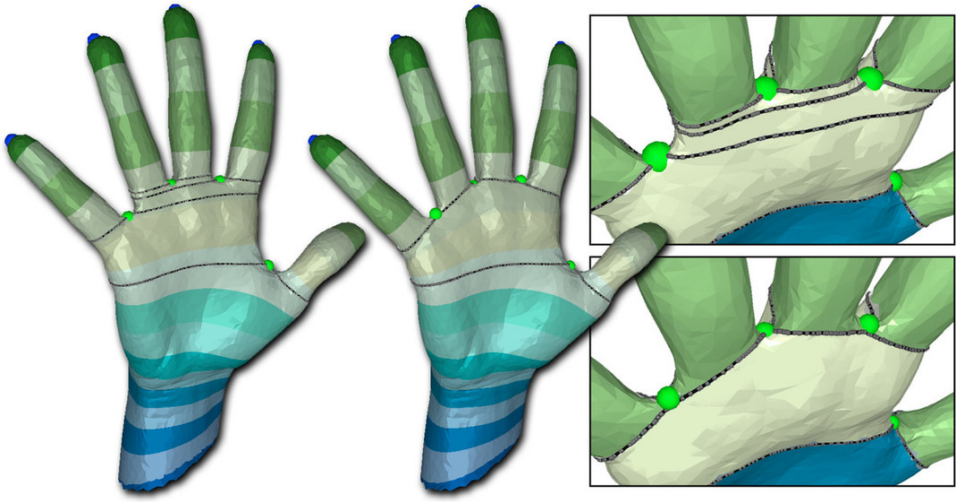


Figure 5.11 – Thin Reeb charts *(left and top) result where multiple saddles have nearly equivalent scalar values. Our global editing operations support the geometric control of the contours, linking the saddle vertices and removing thin Reeb charts (right and bottom).*

extremum. Finally, note that after each editing operation, the Reeb graph is recomputed globally. Then, the list of Reeb charts which need an update of their individual triangle list is tracked based on the lists of regular vertices of the arcs of the Reeb graph (as described in Figure 5.10, center).

Chart number The number of charts can also be edited by the user through a set of atomic operations for the insertion, deletion, subdivision and merging of Reeb charts.

As illustrated in Figure 5.10 (top), disc-like Reeb charts can be inserted or deleted by inserting or deleting extrema constraints from the Dirichlet boundary conditions.

Reeb charts can also be subdivided. In particular, because the Reeb chart is defined as a collection of contours, splitting a chart into two can be achieved by flagging a particular contour (i.e. clicking on a vertex, Figure 5.10, bottom) and by construction each child chart maintains the topological guarantees of the Reeb atlas segmentation. Reeb chart splitting facilitates alignment of the scalar field, the charts' parameterizations and consequently of the final quadrangulation to surface features.

When the scalar field f admits a succession of nearby saddles (Figure 5.11), it may be desirable to align the associated critical contours. In effect, this functionality coarsens the Reeb atlas by removing thin Reeb charts to align extraordinary vertices in the final quad mesh. The atlas coarsening maintains the total number of saddle vertices while decreas-

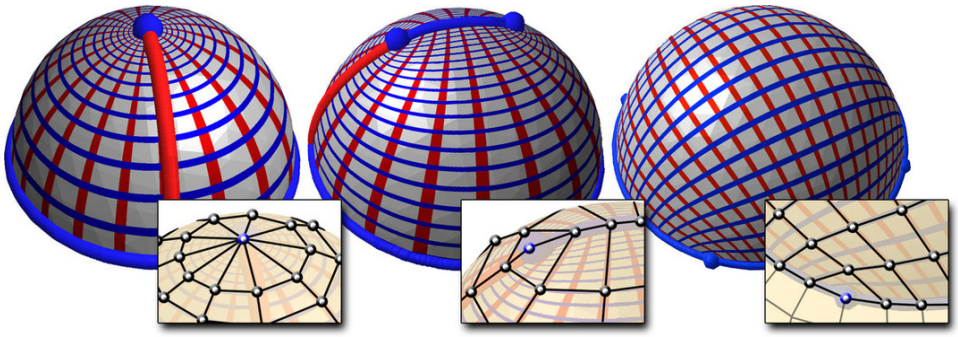


Figure 5.12 – *Fractional poles: a polar vertex (left) split into 2 half- poles (val. 2, middle) and 4 quarter-poles (val. 3, right).*

ing the number of saddle contours (multiple critical points are enforced to share the same function value, forcing f not to be a PL Morse scalar field), yet the Reeb charts remain well defined. Aligning multiple saddles (Figure 5.11) is achieved interactively by first deleting the critical contours to align. Next, the user clicks on pairs of saddles to be connected with automated mesh traversals (shortest paths) providing initial curve segments. Then, the user can further re-orient them with the critical contour widget (the aligned curves are then constrained as discussed previously).

Chart corners The Reeb atlas provides a coarse quadrangular segmentation that can be refined by contouring the parameterization of each Reeb charts and locally solving potential t-junctions as further described in Chapter 7. With this strategy, the corners of each Reeb chart will yield extraordinary vertices (i.e. with a valence different from 4). Here I describe how such vertices can be controlled with atomic edits with the notion of *fractional singularity*.

When a boundary component of a Reeb chart is an extremum vertex, parameterizing the chart with a cutting streamline (as an open annulus) generates a polar singularity that leads to triangular elements around a high valence extraordinary vertex (Fig. 5.12, left). To guarantee the generation of a quad-only output, we use the notion of *fractional singularity*. In particular, our default parameterization strategy for disc charts splits a polar singularity into quarter poles, where the resulting quad mesh contains 4 valence 3 vertices (Fig. 5.12, right).

An alternative proposed to the user is to split the polar singularity into 2 half-poles, constraining a sequence of mesh edges with constant min/max f values (0 or 1); then, the chart is parameterized with a cutting streamline (Fig. 5.12, middle). This configuration corresponds to the concept of *non-isolated critical points* in the smooth setting. We use *Simulation*

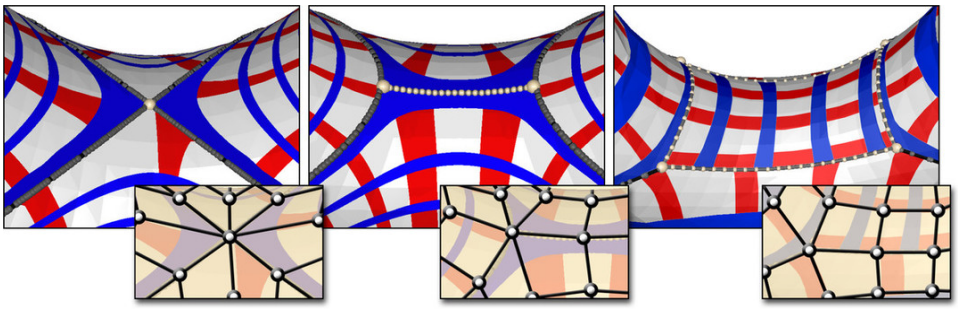


Figure 5.13 – *Fractional saddles: a saddle vertex (val. 8, left) split into 2 half-saddles (val. 6, middle) and 4 quarter-saddles (val. 5, right).*

of Simplicity (*SoS*) (EM90) in the PL setting to maintain a consistent combinatorial representation of f. The resulting quad mesh has 2 valence-2 extraordinary vertices at the endpoints of the extremum segment.

In the spirit of handling fractional polar singularities, we design fractional saddle singularities within the scalar field design. Saddle vertices correspond to extraordinary vertices within the final quad mesh (Fig. 5.13, left). We provide a set of atomic editing operations that enable the user to redistribute easily the high valency of saddles with the notions of *half-* and *quarter*-saddles. While there exists multiple possible combinations of adjacent Reeb chart parameterization configurations, we abbreviate this discussion to the example shown in Fig. 5.13. A non-degenerate saddle contour is a set of two closed curves admitting exactly one common point. Half-saddle splitting is supported by modifying the geometry of the saddle contour to be described, for example, with two closed curves linked by a *middle segment* that is aligned to the edges of the mesh. The half-saddles are defined at the intersection of the middle segment and the two closed curves (Fig. 5.13, middle).

To design half-saddle configurations (Fig. 5.13, middle), the user deletes the original saddle contours and vertex, then initiates the tracing of two contours from manually chosen vertices. The middle segment is automatically computed as the shortest path defined along mesh edges between the two points. User-defined half-saddle contours can be geometrically edited via the critical contour widget to align to surface features. The network of critical contours defining the half-saddle is assigned a single constraint isovalue. Note, the middle segment relates to the notion of non-isolated critical point in the smooth setting, handled in the PL setting with *SoS*. Splitting a saddle reduces the valence of the related vertex by redistributing it among the multiple, created extraordinary vertices. The quarter-saddle configuration (Fig. 5.13, right) further supports this obser-

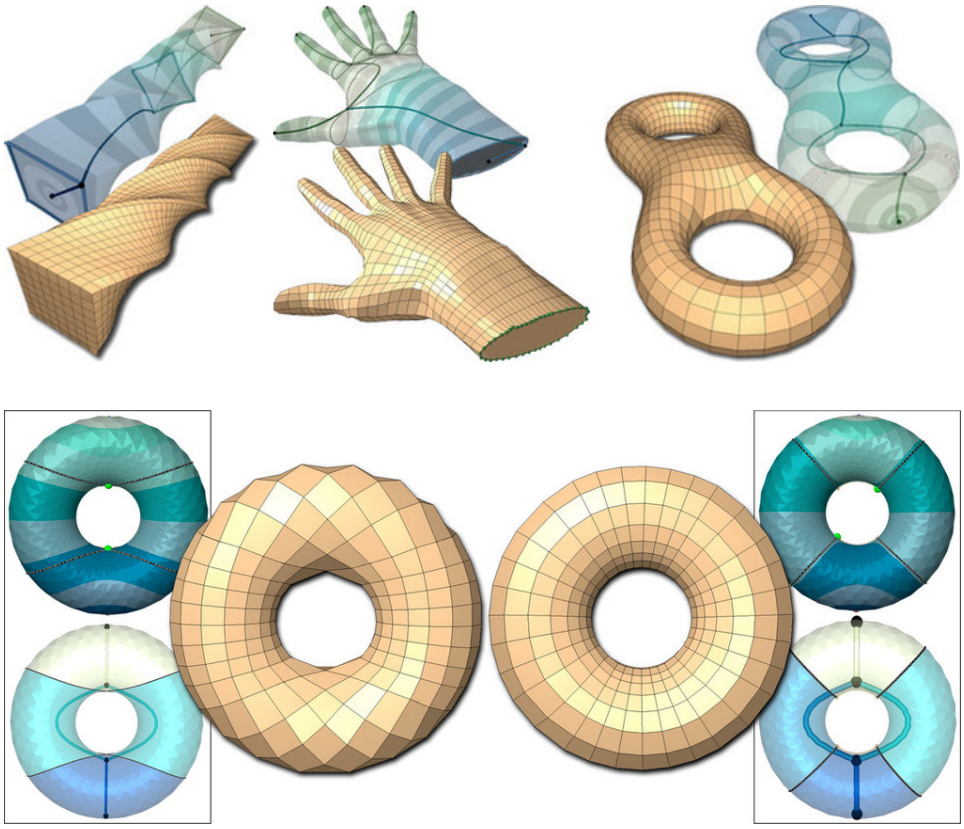


Figure 5.14 – *Preliminary quadrangulation results obtained after Reeb atlas editing and a direct contouring of the Reeb chart parameterizations.*

vation. Quarter saddles are designed by first deleting the original saddle contour and vertex. Then the user clicks on a reference vertex to extract its isocontour. Three other reference vertices are selected along this isocontour and pairs of reference vertices are connected through shortest path computations (Fig. 5.13, right). Finally, an extremum is inserted at the location of the original saddle to maintain a valid field topology. The user can use the critical contour widget to further align the contour (constrained as discussed previously.

Preliminary results Figure 5.14 illustrates a few preliminary quadrangulation results obtained after Reeb atlas editing and a direct contouring of the Reeb chart parameterization. In particular, the hand example (center, top) has been obtained after saddle alignment and chart subdivision at the finger tips. The torus example (bottom) illustrates the effects of fractional singularity editing (left: before, right: after). In particular, due to our default parameterization method for discs, splitting polar vertices into quarter-poles, the construction of half-saddles can lead to the removal of pairs of extraordinary vertices in the quad mesh. In particular, the singu-

larities of *min/split-saddle* and *merge-saddle/max* Reeb charts are removed. On the torus examples, all singularities can be classified as these types, resulting in a completely regular quadrangulation (right). The bitorus example (top right) further exemplifies fractional saddle editing.

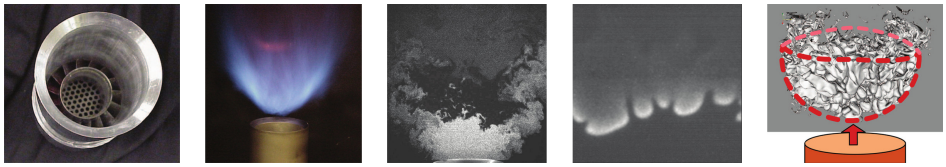
Concluding remarks In this subsection, I described an interactive framework for the complete editing of the geometry and topology of a surface segmentation derived from the Reeb graph, with a specific application to surface quadrangulation. Based on these algorithms, this latter application will be further discussed in Chapter 7, illustrating another type of applications of topological data analysis beyond scientific visualization. In that latter context, our Reeb atlas interactive framework opens new possibilities to incorporate users knowledge into segmentation tasks, as described in the previous subsection with the Morse-Smale complex but this time for segmentation tasks where features of interest align with level sets.

Analysis

6

Contents

This chapter describes my contributions for the quantitative analysis of scientific data based on topological methods. Starting from precise application problems, I describe how these algorithms can be adapted to conduct interactive data exploration and quantitative analysis. First, I describe how the split tree can be used to extract, enumerate and track frames through time in turbulent combustion simulations. While this approach is accompanied with an exploration user interface capable of tracking individual flames, I describe how this approach can also be used to derive quantitative measurements helping in the interpretation of the simulation. Second, I describe how the segmentation capabilities of the join tree and the Morse-Smale complex can be combined to analyze covalent and non-covalent interactions in molecular systems. Such an approach enables not only to robustly extract these features, but also the atoms involved in each localized interaction. For simple systems, our analysis corroborates the observations made by the chemists while it provides new visual insights for larger molecular systems. This chapter presents parts of the results described in the following papers: (BWT*09, BWT*11, GABCG*14).



(a) (b) (c) (d) (e)

Figure 6.1 – *Turbulent combustion applicative context (from left to right): (a) Photo of a typical laboratory low-swirl nozzle; (b) Photo of a lean premixed CH_4 low-swirl flame; (c) Experimental Mie scattering image of a lean premixed H_2 flame; (d) Planar laser-induced fluorescence data imaging the OH concentration in a lean premixed H_2 flame; (e) Rendering of the burning cells of the SwirlH2 simulation data. The cells form a bowl shaped structure with the arrow indicating the direction of the fuel stream.*

6.1 Exploration of turbulent combustion simulations

In this section, I describe an analysis framework for large-scale time varying combustion simulations based on hierarchical split trees. This data abstraction enables a massive data reduction (two orders of magnitude) while still abstracting the relevant information for the representation of the features of interest. This reduction enabled the design of a user interface for the interactive exploration of the features of the simulation, yielding new visual insights later confirmed by quantitative analysis.

6.1.1 Applicative problem

Low-swirl injectors (BC95, POB*07, Che95a, NPB*07, MC08) are emerging as an important new combustion technology. In particular, such devices can support a lean hydrogen-air flame that has the potential to dramatically reduce pollutant emissions in transportation systems and turbines designed for stationary power generation. However, hydrogen flames are highly susceptible to various fluid-dynamical and combustion instabilities, making them difficult to design and optimize. Due to these instabilities, the flame tends to arrange itself naturally in localized cells of intense burning that are separated by regions of complete flame extinction.

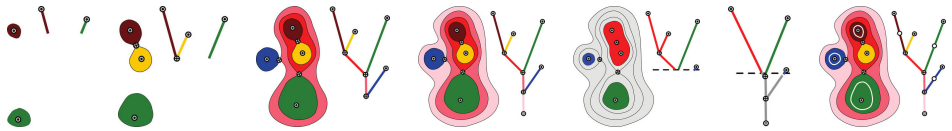
Fig. 6.1(a) shows the detail of a low-swirl nozzle. The annular vanes inside the nozzle throat generate a swirling component in the fuel stream. Above the nozzle the resulting flow-divergence provides a quasi-steady aerodynamic mechanism to anchor a turbulent flame. Fig. 6.1(b) illustrates such a flame for a lean premixed CH_4-air mixture (the illustration shows a methane flame since H_2 flames do not emit light in the visible spectrum). Figs. 6.1(c),6.1(d) show typical experimental data from laboratory low-

swirl, lean H_2-air flames. Such data is used to extract the mean location and geometrical structure of instantaneous flame profiles. The images indicate highly wrinkled flame surfaces that respond in a complex way to turbulent structures and cellular patterns in the inlet flow-field.

Existing approaches to analyze the dynamics of flames, including most standard experimental diagnostic techniques, assume that the flame is a connected interface that separates the cold fuel from hot combustion products. In cellular hydrogen-air flames, many of the basic definitions break down: there is no connected interface between the fuel and products, and in fact there is no concrete notion of a "progress variable" that can be used to normalize the progress of the combustion reactions through the flame. As a consequence, development of models for cellular flames requires a new paradigm of flame analysis.

The computational model used to generate the simulation results explored in this study incorporates a detailed description of the chemical kinetics and molecular transport, thus enabling a detailed investigation of the interaction between the turbulent flow field and the combustion chemistry. As in the physical device, the low-swirl burner simulation achieves a statistically stationary flame in a time-dependent turbulent flow field above the inlet nozzle. Results from the simulation are in the form of a sequence of snapshots in time of the state data. We considered two simulations (labeled SwirlH2 and SwirlH2Fast) having different flow profiles. The SwirlH2Fast case has a mean fueling rate of 2.5 times that of the SwirlH2 case. In the simulations, the time-dependent integrated inventory of fuel in the domain is used to monitor the developing flame. Once a quasi-steady configuration is obtained, snapshots were collected at intervals of approximately 2ms and 1ms for SwirlH2 and SwirlH2Fast, respectively and used for the analysis here. The data sets consist of 332 and 284 snapshots for the slow and fast version, respectively, at an effective resolution of 1024^3. The resulting snapshots are roughly 12–20 Gigabytes in size totaling a combined 8.4 Terabytes of raw data.

The main features of interest are the intensely burning cells defined by a threshold on the local fuel consumption rate. All regions with a local fuel consumption rate above this threshold are tagged as "burning." Note, however, that no single "correct" threshold exists, requiring that we characterize the influence of this threshold value on the resulting diagnostics. However, previous approaches for the analysis of this type of data relied on the identification of a single consumption rate threshold and no approach was available to interactively explore through time and character-



(a) (b) (c) (d) (e) (f) (g)

Figure 6.2 – *(a)-(d) Constructing a split tree and corresponding segmentation by recording the merging of contours as the function value is swept top-to-bottom through the function range. (e) The segmentation for a particular threshold can be constructed by cutting the split tree at the threshold, ignoring all pieces below the threshold and treating each remaining (sub-)tree as a cell. (f) The segmentation of (e) constructed by simplifying all saddles above the threshold. (g) The split tree of (d) augmented by splitting all 1-simplices spanning more than a given range.*

ize quantitatively the geometry of burning flames for interactively defined thresholds. The work described in this section addresses this issue.

6.1.2 Algorithm

The cellular regions of intense burning are identified by thresholding the local consumption rate and we work directly with the resulting three-dimensional, time-dependent regions. The process may be regarded as a generalized subsetting strategy, whereby subregions of a computational result may be sampled, explored and categorized in terms of a volume of space with an arbitrary application-specific definition for its boundary. Since we are interested in regions of high fuel consumption we have identified split trees which encode the topology of sur-level sets, see Chapter 3, as appropriate data structure.

Data segmentation

As discussed in Section 3.2.3, the Reeb graph and its variants (the join, split and contour trees) induce data segmentations with boundaries aligning with level sets, by considering the pre-image through ϕ of each of their 1-simplices. In this application, since burning flames are defined as connected components of space whose combustion rate is higher than a queried threshold $t \in \mathbb{R}$, we follow and extend this segmentation strategy in the case of the split tree.

Given a threshold, t, for the fuel consumption rate $f : \mathcal{M} \to \mathbb{R}$, we determine the corresponding burning cells by conceptually cutting the split tree of f at t creating a forest of trees. Each individual tree represents one connected burning cell, see Fig. 6.2(e). In practice, rather than cut-

ting the split tree and traversing sub-trees, the same information is stored more efficiently as a simplification sequence. A split tree is simplified by successively merging leaf branches with their sibling branch. We order these simplifications by decreasing function value of the merge saddles and store the resulting simplification sequence. In this framework, burning cells at threshold t are defined as sets of all leaf branches with function value greater than or equal to t of the tree simplified to t, see Figure 6.2(f).

Storing only the split tree, the scheme described above allows us to determine the number of burning cells for all possible thresholds. However, in practice we also need an accurate representation of cell geometry and of any number of additional attributes such as volume (see below). Using only the original segmentation this is difficult since we must exclude the lower portions of branches intersecting the threshold. As the distribution of branch attributes can, in general, not be predicted, excluding a portion of a branch would require us to access the original data at significant cost. Instead, we augment the split tree with additional valence two nodes by splitting all branches longer than some threshold, as seen in Fig. 6.2(g). Furthermore, we also compute a number of additional attributes for each branch. For example, we compute the volume of each branch as well as a number of k-th order moments such as means and variances for any variable of interest (not necessarily just f). This splitting is performed with little overhead during the initial computation and allows us to approximate cutting at any threshold with a pre-defined accuracy. It is important to note that this approximation only affects the geometry of the segments and their attributes but not their structure. We are guaranteed to not erroneously merge or split cells due to the approximation.

The augmented split trees form the fundamental data structure in our framework, storing the one-parameter family of possible segmentations along with an arbitrary number of attributes. For efficient access during the visualization we store the segmentation information, a list of vertices per cell, separately.

Feature tracking

Given the data segmentations for all time steps, we track over time features defined by a given static threshold. We track features by spatial overlap which appears to be adequate for our purposes. However, since we have a complete description of features the framework can be easily extended to any number of more sophisticated techniques. To determine

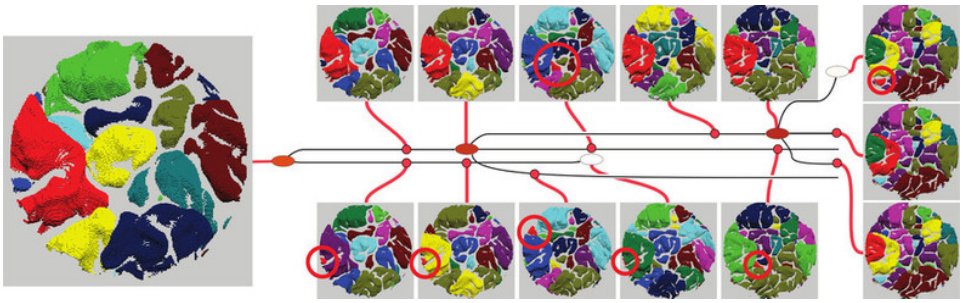


Figure 6.3 – *Example of burning cells being tracked over time (in red). The graph shows a small portion of the tracking graph for the SwirlH2 data set spanning time steps 1880 through 1895. The embedded screen shots show the corresponding segmentation as different nodes are selected in the graph. Over the course of these time steps the large cell represented by the left most node slow sheds smaller cells until it finally breaks apart into three independent pieces.*

the potential overlap we load the split trees of two consecutive time-steps and adapt them to the given threshold. We then traverse the vertices of both segmentations in parallel determining their active segmentation index and if both vertices are above the threshold add a (tracking graph) arc between the corresponding features. The active segmentation index is computed as the segmentation index stored in the file adapted to threshold simplification.

Due to the large number of timesteps involved, creating a tracking graph cannot yet be performed interactively. Furthermore, creating a layout for a given graph, even using state of the art tools, remains too slow for interactive techniques. However, it is important to point out that the tracking graphs are created from the pre-computed segmentations and not the original data so all processing involved can easily be handled by a standard desktop computer.

For regular grids the tracking is by design a streaming process. Each vertex is treated independently and since the vertices are in the same order for both time steps only two consecutive split trees must be kept in memory. For each time interval we dump the partial tracking graph to disk to be assembled at the end.

An example of features getting tracked through time is shown in Figure 6.3. The figure shows a small portion of the tracking graph for the SwirlH2 data set for time steps 1880 through 1895. The embedded screen shots show the main segmentation display when the indicated node has been selected.

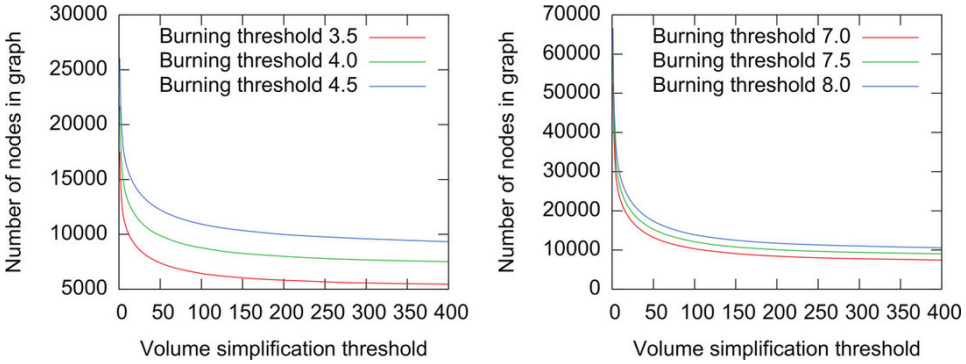


Figure 6.4 – *Determining the tracking graph simplification threshold: The graphs show the number of nodes remaining in the tracking graphs vs. the simplification threshold for the restricted portions of the SwirH2 (left) and SwirlH2Fast (right) data set.*

Tracking graph simplification

As illustrated in Figure 6.3, the tracking graphs can become highly complex and difficult to understand. Furthermore, they contain artifacts of the thresholding such as tiny features existing for only one or very few time steps. To reduce the graph complexity and eliminate some of the artifacts we simplify the tracking graphs by removing all valence zero nodes as well as nodes with a volume smaller than a given threshold (in practice we use the number of vertices corresponding to each node as a measure of volume). Such simplification significantly streamlines the tracking graph by suppressing unnecessary details. In order to avoid disconnecting segments above the volume threshold and thus structurally changing the tracking graph we restrict the simplification to successively removing leafs below the volume threshold. To choose an adequate volume threshold we study how the number of nodes in the tracking graph changes as we increase the volume threshold, see Fig. 6.4. These plots indicate a separation between noise and features. To reduce the complexity of the graphs and the cost of the layout we chose values on the upper range of the suggested noise level for simplification. All examples shown here use a threshold at around 100 vertices.

6.1.3 Results

The data reduction with our hierarchical split tree representation was performed in parallel on an SGI Altix 350, with 32 Itanium-2, 1.4 GHz processors using one processor per time step. The interactive exploration of the data and its quantitative analysis was carried out on a commodity desktop computer. For a single representative time step, the computation of

the hierarchical split tree and corresponding segmentation took 1067 and 2684 seconds for the SwirlH2 and SwirlH2Fast data-sets respectively (excluding I/O). The hierarchical split trees pre-process the data with respect to one of the most important aspects of the data (the burning cells) and store all additional information in accordance with this segmentation, allowing for an interactive post-exploration. Even for the largest data sets, the resulting split trees consist of only around 6Mb ASCII information per time step and their corresponding segmentation to 144Mb, compared to several Gigabytes of raw data. In fact, the trees are small enough to be loaded interactively from disk. Using standard gzip compression, these reduce to roughly 70Mb. Overall, for all time-steps, our data representation roughly required 13Gb and 20Gb of gzipped files for the SwirlH2 and SwirlH2Fast case, respectively. Given the 3.9 and 4.5 Terabytes of original data, this corresponds to a data reduction of more than two orders of magnitude, while still providing greater flexibility in the segmentation and selection than possible using standard techniques based on isosurfacing for instance.

Interactive exploration interface

The primary focus of this work was to provide the application scientists with the ability to comfortably explore their data in a manner meaningful in their particular problem space. For example, allowing the user to explore easily variables conditioned on the extracted features provides a simple way to understand whether various conditional statistics may provide new insights into the data.

Graph display Our user interface presents a fully linked system in which the user can explore the tracking graph, the corresponding segmentation, and the conditional statistics simultaneously with on-demand data loading, as illustrated in Figure 6.5. One of our two main windows (V) is dedicated to the display of the tracking graph (layout with *dot*.) To reduce the visual clutter, only the non-valence two nodes of the tracking graph are shown while sequences of valence two nodes are indicated by unbroken arcs. For exploration we typically use the cell volume (represented by the number of vertices within the cell) to highlight larger cells. To display the graph, we load its geometry into OpenGL, which allows us to draw even the largest graphs fully interactively. The graph display not only provides a visualization of the graph but the user can also select

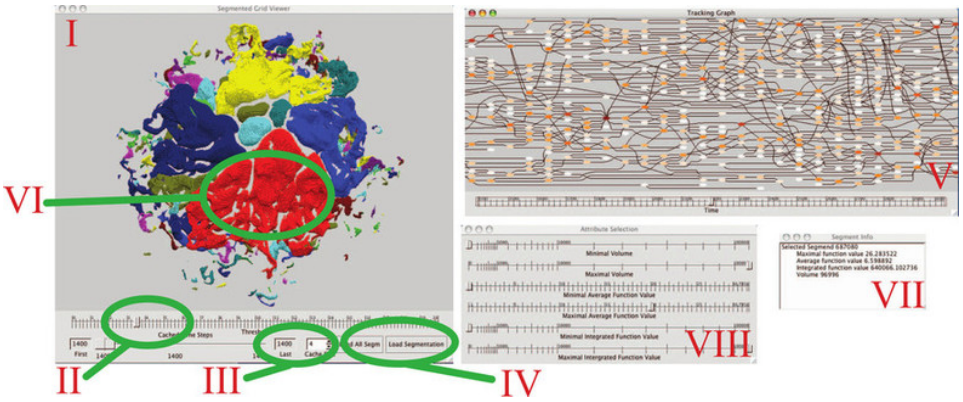


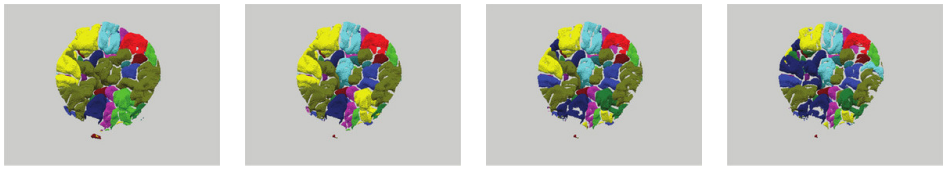
Figure 6.5 – *Illustration of the different components of the user interface. (I) 3D display of the segmentation including a slider to select the fuel consumption threshold (II); (III) the interface to determine the number of in-memory time steps; (IV) the button to load the geometry; (V) interactive display of the tracking graph. Selecting node in either the 3D viewer or the graph display causes the corresponding cell to be highlighted (VI) and its attribute information to be displayed in the info window (VII). The last window (VIII) provides the ability to sub-select segments based on attribute values.*

nodes or arcs. When selecting an arc, the system automatically selects the closest valence two node along this arc. A selection triggers two actions. First, the system loads the split tree of the corresponding time step and, if desired, a number of neighboring time steps. Since split trees are comparatively small, the trees are loaded interactively from disk without any caching or other acceleration mechanism. Second, the segment id and all its corresponding attribute information are extracted from the split tree and displayed in the info window (VII).

Note that the current threshold of the split tree is driven by the segmentation display (slider II), while the time tracking graph uses a single fixed threshold. Thus, during selection, the tracking graph and the split tree can use different thresholds, in which case the system automatically adapts the selection: If the split tree threshold is smaller (bigger cells) the segment containing the picked one is selected; If the split tree threshold is larger (smaller cells) the information for the appropriate sub-segment is shown. Finally, if the node the user has selected corresponds to any segment currently shown, this segment will be highlighted (VI).

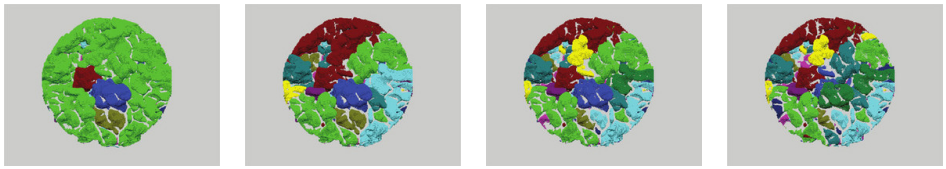
Segmentation Display The other main window (I) displays the segmentation and allows the user to vary the threshold (II) and pick the number of in-memory time steps (III). Individual cells are displayed using one of eleven colors at random, reserving bright-red for highlighted cells.

Similar to the graph display, the segmentation view supports selection



(a) (b) (c) (d)

Figure 6.6 – *Burning cells in the center of the SwirlH2 data at time step 1500 (randomly colored) using fuel consumption thresholds of 4.0 (a), 5.0 (b), 6.0 (c) and 7.0 (d) kg_{H_2}/m^3s respectively.*



(a) (b) (c) (d)

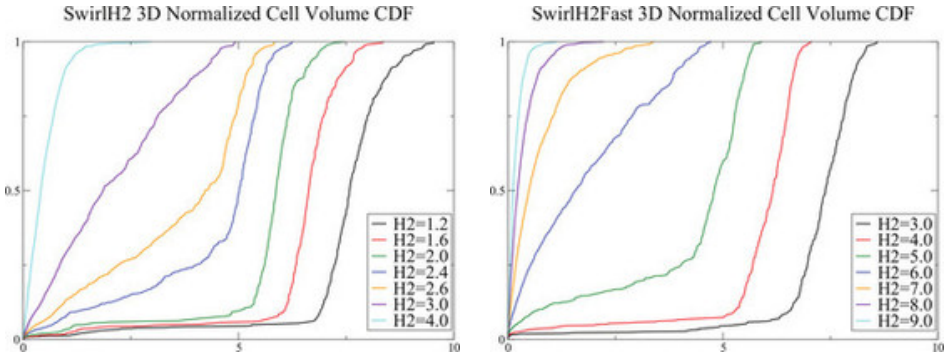
Figure 6.7 – *Burning cells in the center of the SwirlH2Fast data at time step 3000 (randomly colored) using fuel consumption thresholds of 6.0 (a), 7.0 (b), 8.0 (c) and 9.0 (d) kg_{H_2}/m^3s respectively.*

of individual segments displaying their information in a separate window (VII). Finally, we provide an additional window (VIII) that makes it possible to sub-select segments based on the various attributes. Overall, the system supports to exploring the entire time series of a combustion simulation at arbitrary thresholds and using conditional selection criteria.

Quantitative analysis

The flexible, one-parameter families of segmentations generated by our framework and their corresponding statistical information enable new in-depth analysis capabilities in studying turbulent flames. I report in this section quantitative analysis scenario derived from our framework as well as the preliminary insights our collaboration partners gained with these.

The main significant observation is that the flames in the low-swirl configuration seem to burn in two different modes. Overall, the burning cells create a bowl shaped structure centered above the burner. Around the center of this bowl, cells appear to behave much like the idealized flames studied in (DBB*09). On the outside, however, the flames burn more chaotically in smaller, irregularly shape regions. The behavior of these *fringe* cells is very unlike that of the idealized flames and it is not yet clear how to model them. Therefore, the initial analysis has focused on the center of the bowl.



(a) (b)

Figure 6.8 – *Cumulative density functions of the distributions of cell sizes for various fuel consumption thresholds for the SwirlH2 (a) and SwirlH2Fast (b) data sets. Unlike the idealized flame (DBB*09), these distributions indicate few large cells for lower thresholds than the expected many small cells.*

Our interactive exploration user interface helped our collaboration partners identifying a better and refined fuel consumption rate than used in previous studies. In particular, the threshold $2.6kg_{H_2}/m^3s$ used previously exhibited under segmentations yielding only one large region for the SwirlH2Fast data-set. Our interface, with its visual exploration and statistical measure capabilities, helped them refine these thresholds to $5kg_{H_2}/m^3s$ and $8kg_{H_2}/m^3s$ for the SwirlH2 and the SwirlH2Fast data-sets respectively (Figures 6.6 and 6.7).

To confirm these insights, we generated the time tracking graphs for various fuel consumption thresholds and reported the distribution of the size of the burning cells maintained in the graph in Figure 6.8. As suggested by the visualization, the distributions show a markedly different behavior for lower fuel consumption thresholds. For small thresholds the distributions become exponential indicating a small number of larger cells rather than the logarithmic behavior seen in previous studies. However, as the threshold increases, the distributions continuously change to a logarithmic shape. Combining the visual observations of Fig. 6.6 and 6.7 with the statistical results shown in Fig. 6.8 might suggest that the swirling flames behave similar to the idealized flames but at much higher fuel consumption rates.

Overall, it appears that the low-swirling flames are in a substantially different regime than the idealized flames. Our interactive framework coupled with the data analysis made possible by using augmented hierarchical split trees has been instrumental in trying to better understand the underlying dynamics controlling the low-swirling flames. It has open sev-

eral new research directions for our collaboration partners, as improved fuel consumption parameters have been identified in this study to refine their theoretical models.

Concluding remarks

While we can compute the tracking graphs for the full data including the cells on the fringes, the resulting graphs are difficult to handle. Dot currently does not scale gracefully to these large graphs and creating a layout can take hours or fail all together. Furthermore, assuming a layout is created the resulting graphs are difficult to interpret even after heavy simplification. Currently the graphs, unlike the segmentations, are computed for a static threshold. The data structures contain sufficient information to create graphs efficiently for variable thresholds. However, to view these graphs would require an interactive layout, which is beyond the current state of the art. Thus, new paradigms are needed to handle such graphs potentially involving more sophisticated simplification and hierarchical representations.

In conclusion, this work nicely exemplified the three main steps of data analysis and visualization introduced in Chapter 1: abstraction, interaction and analysis. By using hierarchical split trees, our approach enabled an important data-reduction (two orders of magnitude) while still abstracting the relevant information for the representation of the features of interest of the simulation. This massive data reduction made it possible to derive a user interface for the interactive exploration of the features of the simulation, yielding new visual insights later confirmed by quantitative analysis. While this approach has been applied to turbulent combustion simulations, it could be in principle used in any application where a generalized data subsetting (at interactively defined thresholds) and tracking is needed.

6.2 Quantitative analysis of molecular interactions

In this section, I present an approach that combines the segmentation capabilities of the join tree and the Morse-Smale complex for the robust extraction of subtle features of interest (partly aligned with the gradient of a field and the level sets of another) in chemical simulations.

6.2.1 Applicative problem

The chemical properties of a molecular system are mainly governed by the interactions between the composing atoms. In particular, an interaction type of special interest is covalent bonding, which describes the sharing of electrons in between atoms. *Covalent bonds* have been widely studied since the early twentieth century (Lew16). They give rise to the chemical structure of a molecular system, and are deeply investigated in molecular chemistry (Pau60). When dealing with complex molecular systems, a second type of interactions governs many chemical phenomena: *noncovalent interactions*. These are responsible for the bonding between several molecules and the folding of single molecules onto themselves. Examples of chemical processes driven by such interactions include the bonding between a protein and a drug, a catalyst and its substrate, or self-assembly materials. Thus, the understanding of these interactions is necessary for the interpretation of many biological processes and chemical design tasks (e.g., pharmaceutics, nano-technology).

In contrast to covalent bonds, noncovalent interactions release much less energy and are characterized by low electron density values and only slight value variations. This challenges their extraction and analysis by solely studying the electron density. Recently, the signed electron density and the reduced gradient have drawn much attention in the chemistry community. These scalar quantities are derived from the electron density and enable a qualitative visualization of the interactions (CdCCG*14, JKMS*10). However, the analysis of these quantities is mainly done manually (CdCCG*14, CGJK*11, JKMS*10, LCGP*13). An automated extraction and characterization of encoded chemical interactions (in terms of the involved atoms) is still an issue, that we address in this section.

Interactions in molecular systems

Molecular interactions govern the structure of chemical systems by establishing attractive and repulsive balances in-between atoms. These interactions vary in strength and type. Here, we provide a brief characterization and highlight some of their properties. We refer the reader to (Pau60) for further details. For the purpose of our discussion, one mainly differentiates between two classes of chemical interactions.

A *covalent interaction* describes a chemical bond between atoms by sharing electrons. Based on electrostatic grounds, one can provide a simple

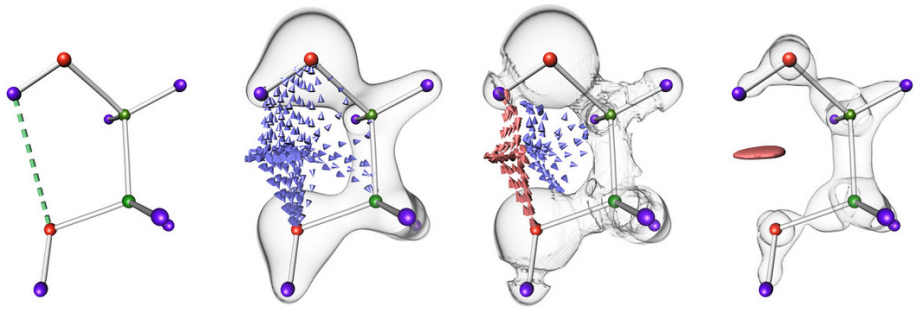
picture of the physics behind it. Placing two atoms next to each-other, the two positively charged nuclei, i.e., the center of the atoms, both attract the outer negatively charged electrons. If the attraction is strong enough such that it overcomes the repulsion caused by the positively charged nuclei, a bond between the atoms is created. The bond represents an energetic equilibrium in which the electrons are equally attracted by the two nuclei. Thus, the two bonded atoms share these electrons. Multiple covalent interactions between atoms result in a system of bonded atoms called *molecule* – typically represented with balls (atoms) and sticks (covalent bonds) (Fig. 6.9(a)).

A *noncovalent interaction* does not involve the sharing of electrons. From an electrostatic point of view, they can be understood as weak electrostatic interactions between temporary and permanent partial charges. While this class of interaction constitutes the driving forces for intramolecular folding and inter-molecular bonding, it spans a wide range of binding energies, which are typically from 1 to 15 kcal/mol and around one to two orders of magnitude smaller than covalent interactions. Hence, their extraction and characterization is much more involved. Noncovalent interactions can be caused by several physical phenomena and we detail here the most relevant ones:

- *Hydrogen bonds:* Among these electrostatic interactions, the hydrogen bonds deserve special attention due to their ever presence in biological systems. They link a hydrogen to an electronegative atom (e.g., oxygen, nitrogen, fluorine, carbon) and occur within the same molecule or in-between molecules. (dashed green line in Fig. 6.9(a)).

- *Van der Waals forces:* These attractive forces have a purely quantum mechanical nature. In particular, the constant movement of electrons around the nucleus transforms an atom into a fluctuating multipole. These temporary charges can cause attraction between close oppositely charged atoms yielding a stable bonding of weak energy. Although the force of an individual van der Waals bond is relatively weak, the cumulative effect of multiple of them may strongly influence the global structure of large molecular systems – as shown in many chemical reactions and protein-ligand interactions.

- *Steric repulsion:* These repulsive forces are short range interactions which occur when two atoms approach one another. Intuitively, they are due to the fact that too many electrons occupy the same



(a) Molecular structure of the 1,2-ethanediol molecule. Noncovalent bonding (green) occurs between the hydrogen and oxygen.

(b) The ρ-field does not capture the noncovalent bond. The gradient flow $\nabla\rho$ (blue arrows) uniformly covers all atoms.

(c) The hydrogen-oxygen attraction is captured by $\nabla\tilde{\rho}$. The attraction (blue arrows) closes a molecular cycle generating a repulsion (red arrows).

(d) The noncovalent bond is given in s by an isolated component (red surface) highlighting the interaction site of the oxygen and hydrogen atom.

Figure 6.9 – *Isosurfaces and gradient behavior (colored arrows) of the electron density ρ, its derived signed electron density $\tilde{\rho}$, and the reduced gradient s for the 1,2-ethanediol molecule. Oxygen, carbon, and hydrogen atoms are shown as red, green, and purple spheres, respectively. Covalent and noncovalent bonds are shown as white sticks and dashed green lines respectively.*

space (Pauli principle). This can be pictured as forces occurring in regions of space bounded by negatively charged elements, such as covalent bonds and negatively charged atoms forming molecular cycles (GCCG*12). The localization of these interactions is of major importance for chemical design tasks since they indicate regions of space that cannot receive additional electrons.

Input data

The input of our analysis are two scalar fields derived from the electron density: the signed electron density and the reduced gradient.

Signed Electron Density In quantum chemistry, electrons behave simultaneously as waves and particules, which only allows for a probabilistic evaluation of their positions. The relative probability that electrons can be found in a particular location of a space is described by the *electron density* $\rho : \mathcal{M} \to \mathbb{R}^+$. Density cusps are expected at the nuclei, the center of the atoms, whereas charges decrease exponentially away from

them. Thus, the nuclei dictate the overall behavior of ρ. Weak atomic interactions are very often occluded and cannot be directly computed or visualized. For instance, while the ethanediol molecule admits a noncovalent bond (dashed green line, Fig. 6.9(b)), this bond is not captured by the electron density ρ (LCGP*13). Investigating the flow of $\nabla\rho$ in Fig. 6.9(b) reveals that the flow enters the molecular cycle from the outside and uniformly covers all atoms forming the cycle. The circular structure shown in Fig. 6.9(a) is not captured by the flow while it is crucial for the analysis of attractive and repulsive interactions. A differentiation of these interactions solely based on the density ρ is not possible, in general. To compensate therefore, a direct investigation of the Hessian $H\rho$ and its eigenvalues is needed (CGJK*11). Assuming the eigenvalues λ_i are given in increasing order, i.e., $\lambda_1 < \lambda_2 < \lambda_3$, we observe the following behavior. In the vicinity of the nuclei all eigenvalues are negative. Away from it, λ_3 becomes positive and varies along the internuclear axis representing covalent bonds. λ_1 and λ_2 describe the density variation orthogonal to this internuclear axis. λ_1 represents the influence of the nuclei, and is always negative away from the nuclei. Contrarily, λ_2 can be either positive or negative depending on the type of interaction. While attractive interactions concentrate electron charge perpendicular to the bond ($\lambda_2 \leq 0$), repulsive interactions cause density depletion ($\lambda_2 \geq 0$). Using this localized information, the *signed electron density* $\tilde{\rho}$ is defined as $\tilde{\rho} : \mathcal{M} \to \mathbb{R}$ with $\tilde{\rho}(x) = sign(\lambda_2(x))\rho(x)$ (CGJK*11). In contrast to ρ which only assesses the interaction strength of atoms, the signed electron density $\tilde{\rho}$ additionally enables the differentiation of attracting and repulsive interactions. Fig. 6.9(c) shows an isosurface of the signed electron density for the ethanediol molecule. In contrast to the electron density, the gradient $\nabla\tilde{\rho}$ captures nicely the attraction between the hydrogen and oxygen (red arrows), which forms a noncovalent bond creating a molecular cycle. This folded conformation also introduces repulsion in the molecule captured by $\nabla\tilde{\rho}$ (blue arrows).

Reduced Gradient To further reveal weak noncovalent interactions, bonds, the *reduced gradient* $s : \mathcal{M} \to \mathbb{R}^+$ of ρ was introduced (JKMS*10)

$$s = \frac{1}{2(3\pi^2)^{1/3}} \frac{|\nabla\rho|}{\rho^{4/3}} \tag{6.1}$$

The reduced density gradient s describes the deviation in atomic densities due to interactions (JKMS*10). Intuitively, covalent and noncovalent

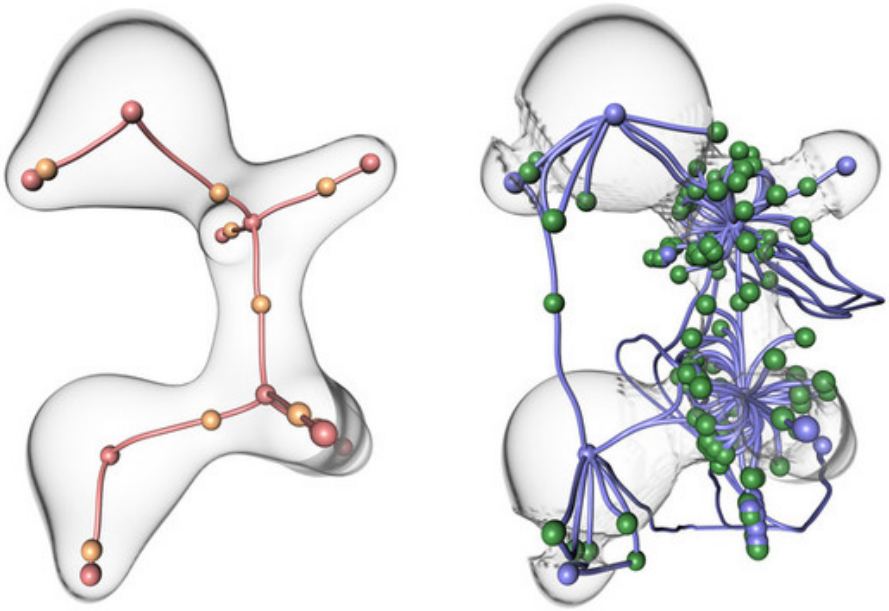
interactions both appear in s. In the presence of such interactions, s reports a strong change in its values in regions of space between interacting atoms. The denominator of s reduces herein the influence of the nuclei due to their high electron densities. In contrast to the electron density, s shows large values in regions far from the nuclei. The electron density exponentially decreases towards zero, and $\rho^{4/3}$ converges faster to zero than $|\nabla\rho|$. Points at which $\nabla\rho$ vanishes become zeros in s and are minima of s. In contrast to the infinite-behavior, the gradient $\nabla\rho$ dominates the denominator $\rho^{4/3}$ at those points.

The reduced gradient provides a qualitative visualization of chemical interactions by considering its isosurfaces chosen at appropriate values (CdCCG*14, JKMS*10). Fig. 6.9(d) shows an isosurface of s for the ethane-diol molecule. The interaction of the hydrogen and oxygen causes the creation of a component of isosurface (highlighted in red). The investigation of such components, which describe regions of space where interactions occur, is the starting point of our approach. In particular, the topological analysis of s reveals *interaction sites* which enables us to focus on locations of space which are relevant from a chemical perspective.

6.2.2 Algorithm

Feature definition

Bader (Bad94) proposed a theoretical model which relates molecular interactions to the critical points of the electron density (MB07). In particular, nuclei can be considered as local maxima of the electron density ρ whereas bond interactions between atoms are represented by separatrices emanating from 2-saddles and ending in two distinct atoms. Fig. 6.10(a) shows these features extracted from the Morse-Smale complex of ρ on the ethane-diol. However, as observed by Lane et al. (LCGP*13), even without any topological simplification, ρ exhibits no 2-saddle in between the upper-left hydrogen and the lower oxygen atoms (cf. dashed line in Fig. 6.9(a)). Thus, no separatrix connecting these two atoms can be extracted and no chemical bond is identified between them. However, as discussed by Lane et al. (LCGP*13), it is established from experimental evidence that this molecule exhibits a noncovalent hydrogen bond connecting these two atoms (Fig. 6.9(a)). Since $\tilde{\rho}$ and s better emphasize noncovalent interactions, we can overcome this limitation by transposing Bader's model from ρ to the analysis of $\tilde{\rho}$ and s. In particular, we focus on the following features.

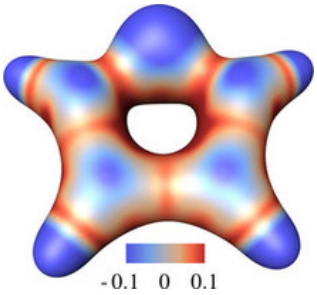


(a) ρ (b) $\tilde{\rho}$

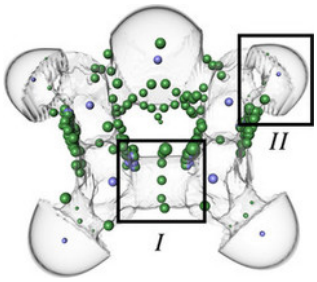
Figure 6.10 – *Critical points (spheres) and separatrices (lines) in the ethanediol molecule for the considered scalar fields. Red, orange, green and blue spheres represent maxima, 2-saddles, 1-saddles and minima respectively. Red and blue lines represent separatrices connecting a 2-saddle to maxima and a 1-saddle to minima respectively.*

Atoms Near the atoms, the flow of $\nabla\rho$ exhibits an attractive behavior characterized by $\lambda_2 < 0$. Thus, $\tilde{\rho}$ is negative in the vicinity of the atoms. Moreover, since the electron population is maximal at the nuclei, $|\tilde{\rho}|$ is also maximal in these locations. Thus atoms are represented in $\tilde{\rho}$ by local minima. With regard to s, these atoms are also represented by minima of s since $\nabla\rho$ vanishes in the atoms. For typical quantum chemical calculations with Gaussian bases, atoms are characterized by $\tilde{\rho} < -0.35$ and $s = 0$.

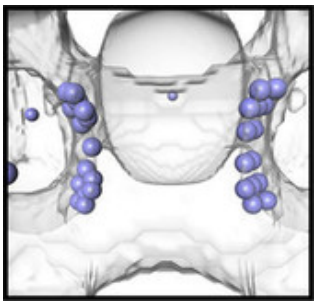
Bond Paths By transposing Bader's model to the analysis of $\tilde{\rho}$, a bonding interaction between two atoms is present if there exists a single line connecting the atoms and which is everywhere tangential to $\nabla\tilde{\rho}$. Since atoms are represented by minima of $\tilde{\rho}$, such a line is represented by two integral lines emanating from minima. Moreover, the existence of such a line between two minima implies the existence of a 1-saddle of $\tilde{\rho}$, which connects the two separatrices emanating from the minima. The latter saddle is called a *bond critical point*. Note that this saddle can be a merging or genus-change 1-saddle.



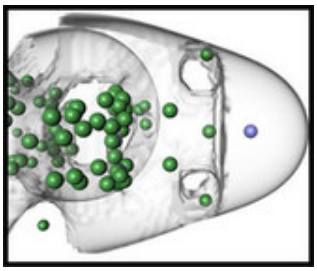
(a) Isosurface of ρ colored by the 2^{nd} eigenvalue λ_2 of $H\rho$



(b) Minima and 1-saddles of $\tilde{\rho}$ scale by persistence.



(c) Zoom-in *I* of (b).



(d) Zoom-in *II* of (b).

Figure 6.11 – *Technical challenges associated with the extraction of molecular interactions from $\tilde{\rho}$ on the furan molecule: beyond low-persistence critical point pairs, $\tilde{\rho}$ also exhibits high-persistence minimum-saddle (c) and saddle-saddle pairs (d) which occlude the critical points of interests (b) representing atoms and bond critical points.*

Bonding Graph The set of atoms and bond paths in a molecular system form the *Bonding Graph*. Each node of this graph represents an atom and its edges represent all attractive interactions in the system.

Repulsive Bond Cycles Bonds can form cycles in the bonding graph yielding steric repulsion. Thus, we define a cycle of minimal length in the bonding graph as a repulsive bond cycle.

Fig. 6.10(b) shows the extraction of the minima and the ascending separatrices ending in 1-saddles, obtained from the Morse-Smale complex of $\tilde{\rho}$. In contrast to the topological analysis of ρ (Fig. 6.10(a)), this analysis reveals a bond path connecting the upper-left hydrogen and the lower oxygen atoms, hence, corroborating experimental observations (LCGP*13). Thus, this result further motivates the topological analysis of $\tilde{\rho}$ for noncovalent bond extraction. Note that s also captures this interaction site by a component of its isosurface (Fig. 6.9(d)).

Technical challenges

Although the Morse-Smale complex of $\tilde{\rho}$ nicely captures covalent and noncovalent bonds, this comes with a price to pay in terms of topological complexity (Fig. 6.10(b)). This complexity occludes the features of interest and challenges their direct extraction, as illustrated in Fig. 6.11.

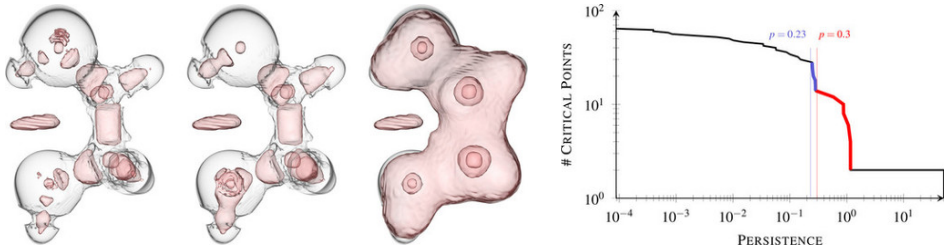
According to our feature definition, atoms can be identified by considering minima of $\tilde{\rho}$. Bond paths can be identified by extracting from the Morse-Smale complex of $\tilde{\rho}$ the two descending separatrices emanating from each 1-saddle. A bond path is considered *valid* if it connects two distinct atoms. Note that the robust extraction of the bond paths is of major importance for the extraction of the remaining features (covalent bonds, noncovalent bonds, Bonding Graph, repulsive bond cycles). However these features are also the most challenging to isolate, for the following reasons:

1. *Low persistence structures:* Critical points are present in the Morse-Smale complex due to the sampled representation of $\tilde{\rho}$. While spurious 1-saddles can yield false positive bond paths, spurious minima can additionally yield interrupted bond paths by preventing descending separatrices emanating from a valid bond critical point to reach the corresponding atoms. Thus, the Morse-Smale complex needs to undergo topological simplification to *(i)* remove low-persistence minima and to *(ii)* connect bond critical points to atoms.

2. *Non-atom minima:* At the edge of attractive ($\tilde{\rho} < 0$) and repulsive ($\tilde{\rho} > 0$) regions (Fig. 6.11(a)), $\tilde{\rho}$ exhibits high-persistence minima, as illustrated in Fig. 6.11(c). These minima also need to be discarded to avoid interrupted bond paths as well as false positives in atom identification.

3. *Non-bonding saddles:* As illustrated in Fig. 6.11(d), $\tilde{\rho}$ also includes high-persistence (genus-change) saddle-saddle critical point pairs. However, this category of critical point pair may not be removable through Morse-Smale complex simplification (JP06). Thus, the removal of these outliers challenges existing topological techniques and motivates the joint analysis of the reduced gradient s.

The above challenges motivate a tailored topological analysis pipeline described in the following section.



(a) Interaction sites obtained from the sub-level set components of the reduced gradient s for three simplification levels: no simplification (left, 32 sites), $p = 0.23$ (middle, 14 sites), $p = 0.3$ (right, 6 sites).

(b) Persistence curve highlighted with two different simplification levels (blue and red vertical lines).

Figure 6.12 – *Illustration of different simplification levels of the reduced gradient s based on the persistence p of critical point pairs. Reading the persistence curve from the right to the left side reveals a first slope (red) at the end of which interaction sites corresponding to covalent bonds have been simplified. The end of the next slope on the left ($p = 0.23$, blue) corresponds to a simplification level where both covalent and noncovalent interaction sites are maintained.*

Interaction sites

To address the third challenge of the previous paragraph, we describe a pre-processing stage whose purpose is to isolate regions of space containing 1-saddles of $\tilde{\rho}$ which are relevant bond critical point candidates from a chemical perspective. Interaction sites can be visualized by considering the regions of space bounded by the connected components of relevant isosurfaces of s (CdCCG*14, JKMS*10). However, each type of interaction is visualized by a specific isovalue. The birth of these components happens at early isovalues for covalent bonds. Contrarily, for noncovalent bonds, the birth happens at arbitrarily larger isovalues. To extract all interaction sites in a single procedure, we analyze $\mathcal{J}(s)$, the join tree of s.

The reduced gradient s exhibits low values in the vicinity of the nuclei and larger values away from it. Moreover, as the isovalue increases, new connected components of isosurfaces appear in regions of spaces where interactions (covalent and noncovalent bonds) occur. The appearance of such components are characterized by minima of s. As the isovalue continues on increasing, the merge of these components occur at 1-saddles of s. These events are captured by $\mathcal{J}(s)$. In particular, each branch of $\mathcal{J}(s)$ which contains a leaf (a minimum) represents a connected region of space containing a single minimum of s. Therefore, we extract each interaction site by considering each branch of the join tree which contains a minimum.

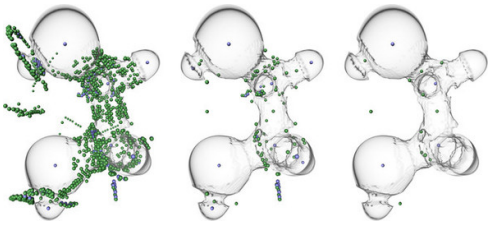
Fig. 6.12 (left) shows the interaction sites (bounded by red surfaces)

extracted from the analysis of $\mathcal{J}(s)$. Due to the sampled representation of s and its numerical evaluation, spurious features are present – in particular in the vicinity of the atoms. To discard these, we apply a persistence based simplification of $\mathcal{J}(s)$. To select an appropriate simplification level, we manually inspect the persistence curve of the saddle-minimum pairs (Fig. 6.12) for the ethanediol molecule. Reading the curve from the right side (high persistence pairs) to the left (low persistence pairs) reveals a first slope (red in Fig. 6.12) finishing with a sharp kink at $p = 0.3$ (red vertical line). This level corresponds to the removal of the minima representing covalent bonds. Thus, at this level of simplification, the interaction sites extracted from $\mathcal{J}(s)$ (Fig. 6.12(a), right) only reveal: the prominent atoms (oxygens and carbons), a larger component which represents the sub-level set containing all covalent interactions, and an isolated component representing a noncovalent interaction. The following slope to the left (blue) also finishes with a sharp kink at $p = 0.23$ (vertical blue line). This point corresponds to a simplification level where the components representing covalent bonds are maintained (Fig. 6.12(a), middle). This observation was confirmed in all of our experiments. Thus, we select as an appropriate simplification level the end of the second slope (from the right side) on the persistence curve. As illustrated in Fig. 6.12(a) (middle), each interaction (covalent and noncovalent) is represented by a specific interaction site. Note that dominant atoms (oxygens and carbons) are also represented. However, the latter regions typically include no 1-saddle of $\tilde{\rho}$ which enables to discard them in a straightforward manner. Note that at a technical level, in contrast to the Morse-Smale complex of $\tilde{\rho}$, all saddle-minimum pairs captured by the Join Tree of s can be simplified, which enables to overcome the technical challenge introduced by the incomplete simplification of the Morse-Smale complex of $\tilde{\rho}$.

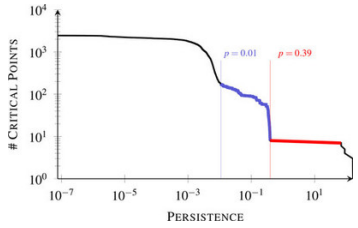
Bonding graph

Covalent and noncovalent bonds form a subset of the More-Smale complex. We describe how the Morse-Smale complex of $\tilde{\rho}$ can be reduced to the Bonding Graph representing all atoms and their bonds only.

1. Removal of Low Persistence Structures First, the Morse-Smale complex of $\tilde{\rho}$ needs to be simplified. To find a representative simplification level, we analyze the persistence p of all minima and saddles. Fig. 6.13(b) shows such a curve for the ethanediol molecule in logarithmic scaling.



(a) Minima (blue) and 1-saddles (green) for three simplification levels: no simplification (left), $p = 0.01$ (middle), $p = 0.39$ (right).



(b) Persistence curve highlighted with two different simplification levels (blue and red line).

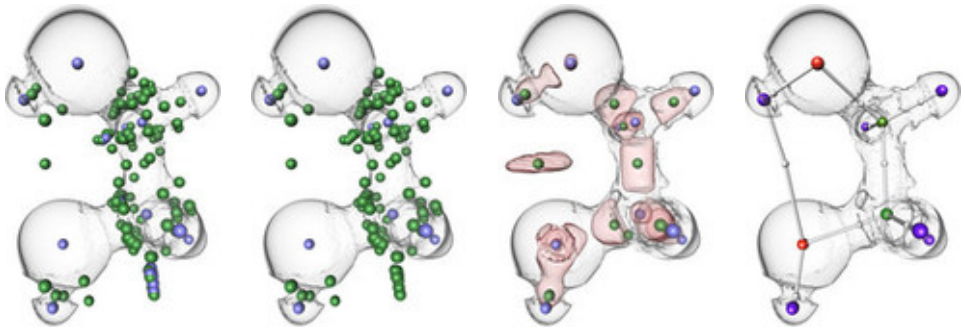
Figure 6.13 – *Illustration of different simplification levels of the signed electron density $\tilde{\rho}$ based on the persistence p of critical points. In the initial level minima and 1-saddles representing atoms and bonds are occluded by low-persistence critical points. Simplifying the Morse-Smale complex removes spurious critical point revealing the features of interest. However, not all minima and 1-saddles represent atoms and bonds even in the simplified Morse-Smale complex.*

The persistence curve exhibits a characteristic behavior. We can make use of this to guide the manual selection of an appropriate simplification level. Around 90% of the minima and saddles have a very low persistence ($p < 0.01$). After the first drop down in this curve, two different persistence-ranges are apparent. In the first range ($0.01 \leq p \leq 0.39$), the curve shows an almost constant slope (blue) indicating the transition from low to high persistent features. Fig. 6.13(a) (middle) shows the minima and 1-saddles at the simplification level $p = 0.01$. Most of the spurious structures are removed at this level, and the critical points representing atoms and bonds are revealed. However, there exists also minima and 1-saddles which do not represent molecular features. Increasing the persistence threshold now iteratively removes pairs of minima and 1-saddles. At the beginning of the second range ($p = 0.39$, red), only the minima representing the heavy carbon and and oxygen atoms are still present. The minima representing hydrogens are already removed due to the small mass of hydrogens. However, 1-saddles which do not represent bonds are still visible in the interface between attractive and repulsive regions. We made the above observation in all of our experiments. To guarantee that all atoms and their even weak bonds are well represented, we select a simplification level after the first drop down in the persistence curve. However, the point of first drop down varies depending on the overall structure and energy of the molecular system.

2. Removal of Non-Atom Minima As detailed previously, there are minima which do not represent atoms. These are characterized by their location at the interface between the attractive and repulsive regions. Thus, their signed density value needs to be larger than those of the atoms. This allows a differentiation of the minima, and we select those with $\tilde{\rho} > -0.35$. Since those minima can interrupt bond paths, we need to remove them from the Morse-Smale complex yielding a new connectivity between the remaining minima and 1-saddles. We follow here the general strategy for the simplification of the Morse-Smale complex, but restricted to the selected set of minima. For each minimum, we determine the lowest saddle in its neighborhood; and put all minima-saddle pairs with their weight, i.e., the height difference between minimum and saddle, in a priority queue. This queue is processed in ascending order starting with the pair with the lowest weight. This pair is removed from the Morse-Smale complex, which requires a subsequent update of the connectivity information of the critical points in the Morse-Smale complex. Due to the new connectivity, the weights of the minimum-saddle pairs also need to be updated yielding a new lowest minimum-saddle pair. We iteratively remove the selected minima until all of them are processed. This yields a Morse-Smale complex in which all minima represent atoms only.

3. Removal of Non-Bonding Saddles Next, only the 1-saddles of $\tilde{\rho}$ which are relevant from a chemical perspective should be considered for bond path extraction. Thus, we restrict the remainder of the analyis to the 1-saddles located in each of the interaction sites extracted previously (Fig. 6.12).

4. Bonding Graph Extraction Each 1-saddle is connected to either one or two minima. Since bonding only occurs between atoms, we neglect all 1-saddles which are twice connected to the same minimum. From the remaining saddles, we collect the 1-saddles S^k and their connected minima M_1^k and M_2^k giving rise to a set of triplets (M_1^k, M_2^k, S^k). It may happen that they exist two saddles S^{k_1} and S^{k_2} which end in the same pair of minima, i.e., $M_1^{k_1} = M_1^{k_2}$ and $M_2^{k_1} = M_2^{k_2}$. However, only one bond can exist between the same pair of atoms. Considering an evolution of the isovalue in $\tilde{\rho}$, bonding critical points are given by the earliest contact of the sub-level set components emanating from the two atoms. Hence, for each interaction site, we choose the 1-saddle S which minimizes its $\tilde{\rho}$-value as the bonding saddle representing the bond between M_1 and



(a) (b) (c) (d)

Figure 6.14 – *Illustration of the different steps of our analysis algorithm: (a) low-persistence pairs are cancelled; (b) high-persistence minimum-saddle pairs are removed; (c) high-persistence saddle-saddle pairs are filtered out based on the interaction site extraction; (d) final bonding graph.*

M_2. Applying this procedure to all saddles S^k yields the Bonding Graph $G = (N, E)$ in which all minima represent atoms and each pair of minima is only connected by a single 1-saddle representing their bonding. The nodes N of the undirected graph G are given by the minima and bonding 1-saddles, its edges E by the separatrices connecting the saddles to the minima.

Fig. 6.14 shows the different steps of our pipeline on the ethanediol molecule and illustrates its ability to isolate the features of interest, despite the presence of high-persistence and spurious critical points in $\tilde{\rho}$.

Repulsive bond-cycles

A repulsive bond cycle is given as a cycle of minimal length in the Bonding Graph. Thus, we need to compute the shortest path between two bonded atoms by omitting their bond, as illustrated in Fig. 6.15. To do so, we weight the edges E of the graph G by the Euclidean distance of the incident nodes. Note that all the weights are positive. Let us consider a node representing a 1-saddle $S \in N$ and its adjacent minima $M_1, M_2 \in N$. The two edges connecting these three nodes are denoted by $\mathbf{E_1}, \mathbf{E_2} \in E$. We apply Dijkstra's algorithm to compute the shortest path $P \subset$

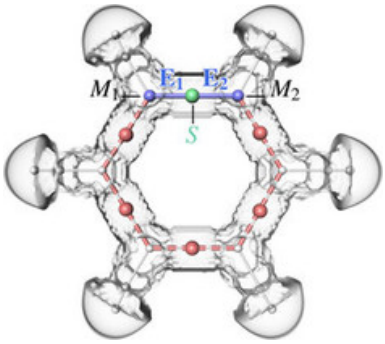


Figure 6.15 – *Illustration of the repulsive bond-cycle extraction.*

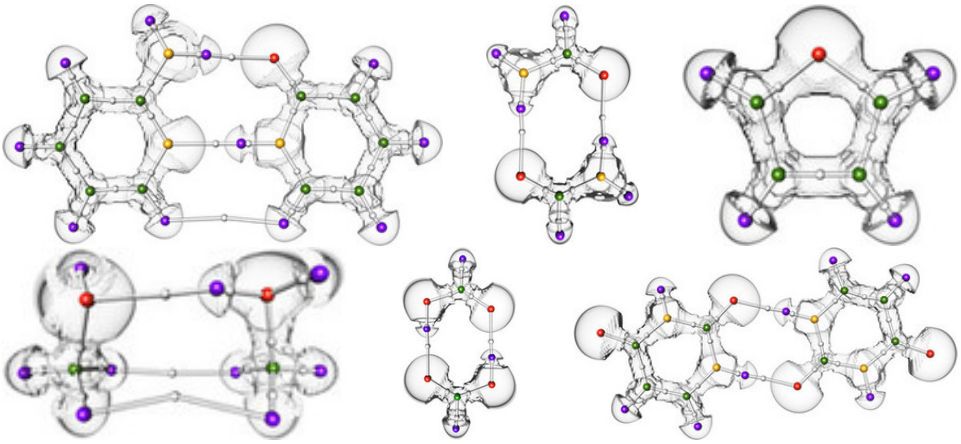


Figure 6.16 – *Bonding Graphs of simple molecular systems. Oxygen, nitrogen, carbon, and hydrogen are shown as red, yellow, green, and purple spheres. Bonding saddles are depicted as white spheres. The Bonding Graph captures covalent and noncovalent bondings (white lines).*

$E \setminus \{\mathbf{E_1}, \mathbf{E_2}\}$ between M_1 and M_2 in G omitting the edges $\mathbf{E_1}$ and $\mathbf{E_2}$. If it exists, a molecular repulsive cycle was found. We mark all minima covered by the shortest path and collect all bonding saddles connecting the minima. We assign to each of these saddles an identifier indicating that all of them belong to the same repulsive interaction. $\mathbf{E_1}$ Note that several saddles can describe the same cycle, i.e., all minima covered by two cycles coincide. In this case, we only keep one representative of this cycle.

6.2.3 Results

This section presents experimental results of our analysis algorithm obtained with a C++ implementation on a computer with an i7 CPU (2.8 GHz) and 16Gb of RAM. We investigate a variety of molecular systems obtained through quantum chemistry simulation with the Gaussian program and represented by regular grid data. In all of experiments, our analysis took in general a few seconds to compute and at most 8 minutes for the DNA helix data-set ($170 \times 178 \times 183$).

Validation

Fig. 6.16 shows the bonding graph computed by our approach on a variety of simple molecular systems. Our analysis indeed reveals the covalent bonds of these systems. Moreover, it also reveals the noncovalent interactions responsible for the bonding of several dimers. In particular, our analysis reveals that two hydrogen bonds (noncovalent bonds linking a hydrogen to a heavier atom) are involved in the bonding of the Pyridoxine

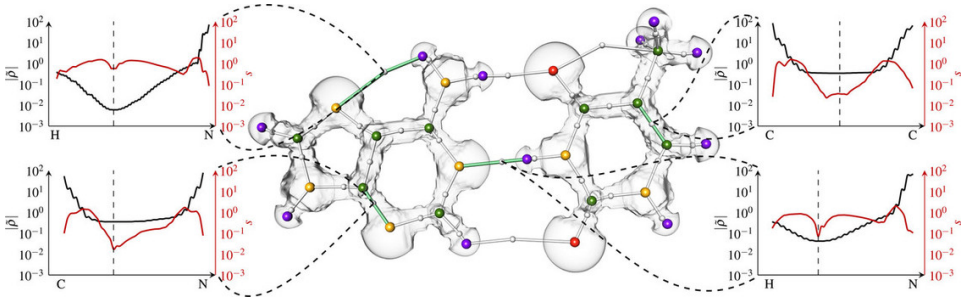


Figure 6.17 – *Quantitative analysis of attractive interactions in the A-T DNA base pair. Thanks to the robust extraction of attractive bonds, each type of interaction can be investigated in the light of several quantum chemistry measures. By restricting the analysis of these measures to the separatrix representing each bond, our analysis enables to project these complex 3D informations down to easily readable 1D plots, revealing distinct characteristic behaviors for covalent bonds (bottom-left, upper-right), hydrogen bonds (bottom-right) or Van der Waals interactions (upper-left). The bond critical points are highlighted as dashed lines in the 1D plots.*

and Aminopyridine (leftmost, top), while a van der Waals attractive force bonds the bottom two hydrogen atoms together. Each of these bonding graphs reveal a covalent and noncovalent bonding structure.

Quantitative Analysis and Visual Exploration

Since our technique allows for a robust extraction of covalent and noncovalent interactions, it also enables to enumerate, classify and investigate such features from a quantitative point of view.

Bond Investigation and Classification In addition to the enumeration of the bonding interactions, our analysis enables new investigation capabilities on a per interaction basis. In particular, one can further analyze various quantum chemistry measures for each extracted bond since our analysis provides a concrete geometrical representation for each interaction. Fig. 6.17 illustrates such an analysis where several quantities ($|\tilde{\rho}|$ and s) were evaluated along the bond paths representing two covalent bonds (lower left and upper right) and noncovalent bonds (upper left and lower right). For each interaction, such a procedure enables to project these complex 3D information down to easily readable 1D plots revealing the evolution of these measures along the bond path. In particular, these plots reveal a characteristic behavior for covalent bonds. A high-valued flat plateau of electron population ($|\tilde{\rho}|$) is located in the middle of the bond. Moreover, s indicates a dominant minimum in the vicinity of the bond critical point. In contrast, noncovalent bonds reveal a more subtle elec-

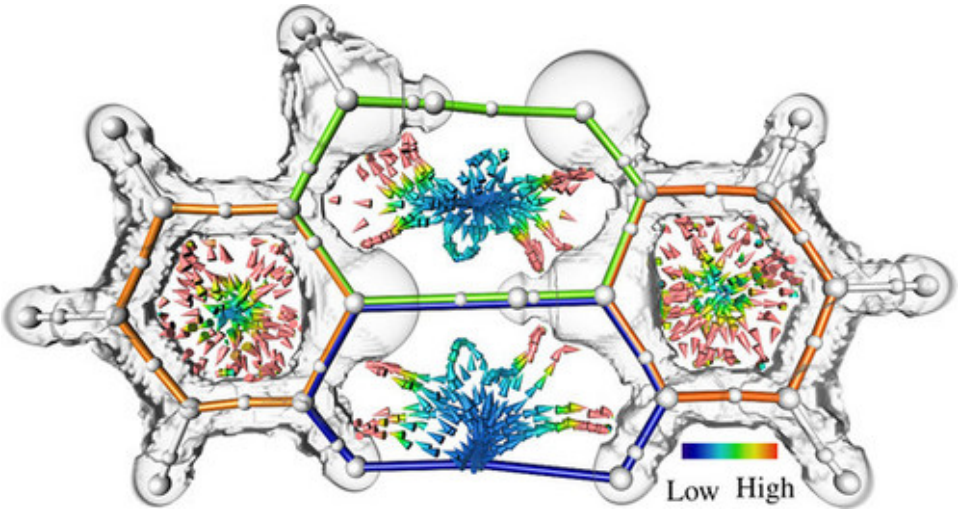


Figure 6.18 – *Repulsive bond cycles of the Pyridoxine-Aminopyridine. Integrating $|\tilde{\rho}|$ along them enables to assess the strength of the steric repulsion, as confirmed visually by the $|\tilde{\rho}|$-color-coding of $\nabla\tilde{\rho}$ (arrows).*

tronic structure. In the vicinity of the bonding critical point, this analysis reveals that a minimum of electron population ($|\tilde{\rho}|$) is achieved at distinct values for hydrogen bonds (10^{-1}, bottom right) and van der Waals (10^{-2}, upper left) interactions; which is one to two orders of magnitude lower than covalent bonds. As proposed by Bader (Bad94), attractive bonds can be classified according to the $|\tilde{\rho}|$-value of their bonding critical point. We make use of this property to classify attractive bonds in the remainder.

Steric Repulsion Fig. 6.18 illustrates the extraction of repulsive bond cycles on the Pyridoxine-Aminopyridine. Steric repulsion is induced by a closed chain of atoms. As a first approach, we integrate $\log(|\tilde{\rho}|)$ along each cycle to assess its strength. This analysis reveals strong steric repulsions induced by the cycles formed by covalent bonds only (left and right orange cycles). In contrast, very weak steric repulsion is induced by the bottom cycle. This difference is caused by the presence of the electrons involved in covalent interactions and their absence in weak-energy van der Waals bond (bottom hydrogens). This insight is also confirmed by the $|\tilde{\rho}|$-color-coding of the numerical streamline integration in $\nabla\tilde{\rho}$. *s*

Complex Molecular Systems Fig. 6.19 shows the extraction of attractive covalent and noncovalent bonds on a folded β-sheet polipeptide. Our analysis enables to classify them according to the $|\tilde{\rho}|$-value of their bond critical points. This classification highlights the noncovalent bonds responsible for the folding of this molecule. While hydrogen bonds (cyan) are in-

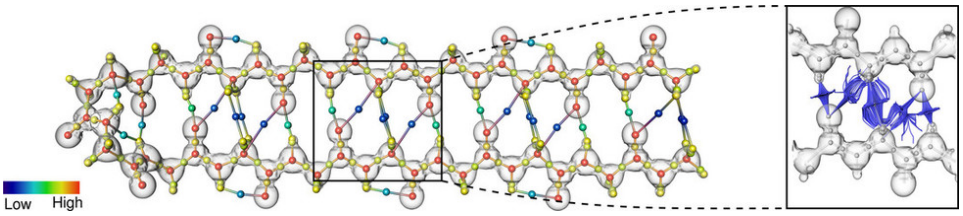
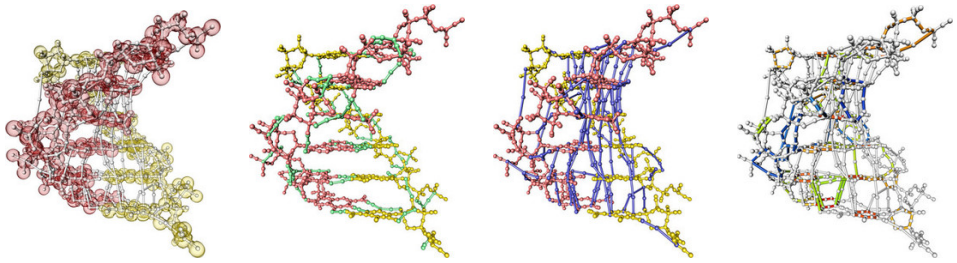


Figure 6.19 – *Visual and quantitative exploration of covalent and noncovalent bonds in the β-sheet polipeptide. The amplitude of the signed electron density ($|\tilde{\rho}|$, color-coded from blue to red) enables to distinguish covalent bonds (yellow) from hydrogen bonds (cyan) and van der Waals interactions (dark blue). While the numerical integration of $\nabla\tilde{\rho}$ (right inset) enables to visually distinguish the latter two types of interactions, our combinatorial pipeline robustly extracts these features to support further quantitative analysis. In particular, our algorithm reveals the repeating pattern (black frame) of noncovalent interactions responsible for the folding of this molecule, which decomposes it in unitary building blocks corresponding to the elementary amino-acids composing the molecule.*



(a) Bonding Graph (b) Covalent and Hydrogen Bonds (c) Covalent and van der Waals Bonds (d) Cycles

Figure 6.20 – *Quantitative exploration of the nucleic acid double helix data-set. Our analysis enables to enumerate and classify noncovalent interactions (Fig. 6.21), yielding query-based visualizations (b, c) revealing the role of each type of interaction in the helicoidal structure of DNA. A multi-scale exploration of the repulsive bond cycles (d) reveals high repulsions in the center of the helix and weaker repulsion on its outer boundary.*

volved in this folding, our analysis additionally reveals that van der Waals attractions (dark blue) also play a structural role to enforce the stability of this folded conformation. This distinction is confirmed by the continuous flow of $\nabla\tilde{\rho}$ (right inset), which exhibits the dispersed behavior of van der Waals attractions. Moreover, as highlighted with the black frame, our analysis reveals a repeating pattern of noncovalent bonds, which corresponds to the decomposition of the molecule in its elementary amino-acids.

Fig. 6.20 illustrates an exploration of the molecular interactions within a portion of a nucleic acid double helix found in DNA. Due to the complexity of this data-set, our algorithm extracts a large number of interactions between the two parts of the helix, highlighted in yellow and red (Fig. 6.20(a)). To reveal the structural role of these interactions at a global

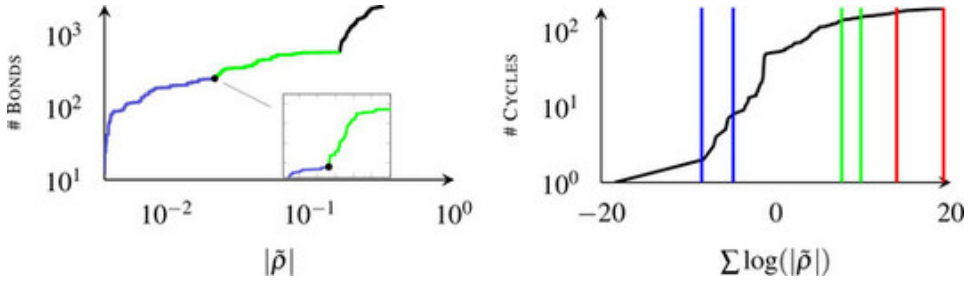


Figure 6.21 – *Supporting plots for Fig. 6.20. Shown are the number of bonds and cycles present in the DNA example as a function of* $|\tilde{\rho}|$.

level, we perform a quantitative analysis of each of the extracted bonds. In particular, plotting the number of bonds as a function of $|\tilde{\rho}|$ (Fig. 6.21 left) indicates three distinct ascending modes. The first two modes (blue and green) are separated by a small-scale kink (see inset) while the last two modes (green and black) are separated by a large-scale kink. As suggested by our initial quantitative analysis on the A-T DNA base (Fig. 6.17), each portion of the curve (blue, green and black) correspond to a specific type of interactions: van der Waals, hydrogen bonds and covalent bonds, respectively. Thus, this curve enables to select thresholds for a query-based exploration of the noncovalent bonds. In Fig. 6.20(b), van der Waals attractions were removed while in Fig. 6.20(c) hydrogen-bonds were filtered out. Covalent bonds appear in red and yellow in these figures. These visualizations provide two complementary global insights on the helicoidal structure of this molecular system. In particular, Fig. 6.20(b) reveals that hydrogen bonds are mostly located in the planes orthogonal to the helix axis while van der Waals attractions follow the axis-direction. This confirms that the torsional stiffness of the double helix, which influences the circularisation of DNA, is mostly governed by hydrogen bonding, whereas the axial stiffness, which characterizes the wrapping properties of DNA, is mostly governed by weaker van der Waals attractions. A similar query-based visualization can be carried out regarding the steric repulsions. Since our analysis enables to quantify the strength of repulsive cycles (cf. previous paragraph), the evolution of their number can be plotted in function of their strength (Fig. 6.21 right). From this curve, the user can select relevant intervals (blue, green, red) and have a direct feedback in the visualization of the corresponding steric repulsions (Fig. 6.20(d)). This enables a multi-scale exploration of such features, which indicates a strong repulsion in the center of the base pairs (red cycles on the horizontal steps of the helix) and weaker repulsion on the outer boundary of the helix (blue cycles).

Concluding remarks

In this work, we showed that subtle features could be reliably extracted in molecular systems by combining the segmentation capabilities of the join tree and the Morse-Smale complex of two scalar fields. While such a tailored approach enables an interactive exploration of the structure of a molecular system as well as its quantitative characterization, it also indicates that the joint topological analysis of pairs of scalar fields may yield more generic analysis algorithms. However, this requires a generalization of topological analysis to bivariate data, as further discussed in Chapter 8.

Related Problems

7

Contents

This chapter briefly reviews my contributions to related problems, where the solution has been derived or inspired from topological data analysis. In particular, I describe an approach for the geometrical design of scalar fields with topological guarantees. Such a procedure is important to certify the solution of a geometrical algorithm. In particular, it has been instrumental in our Reeb-graph based segmentation framework (Chapter 5) to guarantee the topological consistency of the segmentation upon each editing interaction. The application of this framework to surface quadrangulation is also further discussed in the present chapter. Next, in the context of photograph composition for panorama creation, I show how an analysis of the topology of the layout of images enables to derive fast and flexible seam computation algorithms. Finally, I describe an algorithmic framework that enables to assess the accuracy of implementations of isosurface computation (a fundamental task in data visualization and analysis) from a topological point of view. This chapter presents parts of the results described in the following journal papers: (TP12, TDN*12, TDN*11, STP12, PST*15, ENS*12).

7.1 Scalar field design with topological guarantees

Many geometry processing problems involve numerically sensitive tasks such as partial differential equation resolution, gradient field integration, or scale-space computation. In many cases, the topology of the numerical solution is a major consideration. In meshing for instance, extraordinary vertices often correspond to singularities (as discussed in Chapter 5) and these important constraints must be respected. However, numerical noise often occurs and can alter the topology of the solution. We illustrate this issue with the Laplace equation subject to Dirichlet boundary conditions (previously introduced in Chapter 5). Beyond its ubiquity in geometry processing, this equation plays an important role in electromagnetism, astronomy and fluid dynamics. Given a finite set of extrema constraints $\mathcal{D}$ along with corresponding target values, the solution f to this equation is defined as follows:

$$f(d_i) = f_{d_i} \quad \forall d_i \in \mathcal{D} \tag{7.1}$$

$$\Delta f(v) = 0 \quad \forall v \notin \mathcal{D} \tag{7.2}$$

where Δ stands for a discretization of the Laplacian operator on surfaces. An important property of this equation is that the Dirichlet constraints $\mathcal{D}$ should be the only extrema of the solution.

However, since the Laplacian is a second-order operator, it is difficult to discretize for piecewise linear functions. Hence several discretization strategies have been proposed (see (WMKG07) for a comprehensive discussion). Fig. 7.1 shows the solution of this equation for two discretizations of the operator, obtained by least-squares optimization with the penalty method (XZCOX09). The combinatorial Laplacian (WMKG07) (Fig. 7.1(a)) is a straightforward discretization which exhibits robust topological properties. However, as it is strongly biased by the discretization of the mesh, it fails at generating smooth level sets. In contrast, the discretization based on cotangent weights (PP93) produces much smoother level sets. However, in practice, surface triangulations often include many sharp triangles. As edge angles get closer to zero, the numerical error on their tangent evaluation can be arbitrarily amplified when used as the denominator for the cotangent computation, hence yielding an error of arbitrarily high amplitude in the solution in the vicinity of sharp triangles. As shown in Fig. 7.1(b), this numerical error generates additional critical points (with non-zero persistence), which prevents the solution from conforming to its formal description.

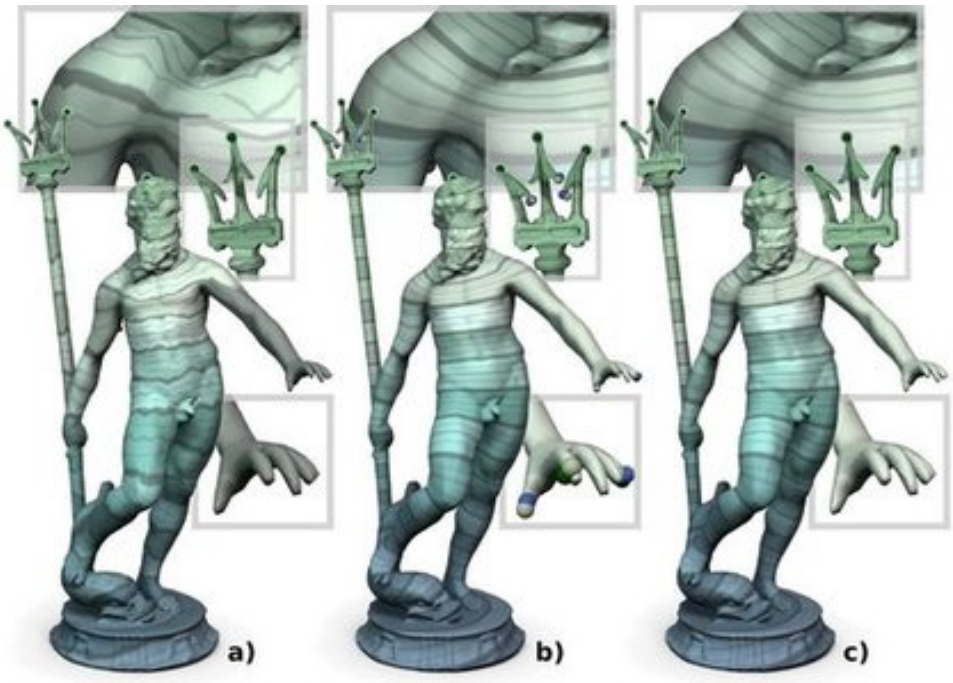


Figure 7.1 – *Solving the Laplace equation with topological guarantees (transparent spheres are Dirichlet constraints). The combinatorial Laplacian (a) satisfies the topological properties of the solution, but it has a poor geometrical accuracy (level lines, top). The cotangent weight Laplacian (b) provides an improved geometrical approximation (top) but is numerically sensitive and generates invalid additional singularities (inset zooms). Our algorithm can be applied as a post-process (c), with no threshold parameter, to remove these inconsistent critical points. The resolution of the equation took 0.14 s. (surface: 25k vertices), while the combinatorial simplification took 0.02 s. (1 iteration). Our algorithm provides a solution (c) which both benefits from the geometrical approximation quality of cotangent weights ($||f - g||_\infty = 0.12\%$) and from the topological stability of the combinatorial Laplacian.*

The generalized topological simplification algorithm that I introduced in Chapter 4 can be used in a straightforward manner to fix these numerical instabilities by using the Dirichlet constraints as topological constraints (C_g^0 and C_g^2). In practice, our algorithm is around an order of magnitude faster than the actual numerical optimization (using CholMod). Thus it can be used as a post-process with a negligible computation time overhead. Note that since our algorithm uses no threshold parameter, it can be used in a robust manner, irrespective of the amplitude of the numerical error. As shown in Fig. 7.1(c), our algorithm automatically removes topological noise while minimally affecting the function. Therefore, our approach can be used to generate a solution with both the geometrical accuracy of the cotangent weight Laplacian and the topological robust-

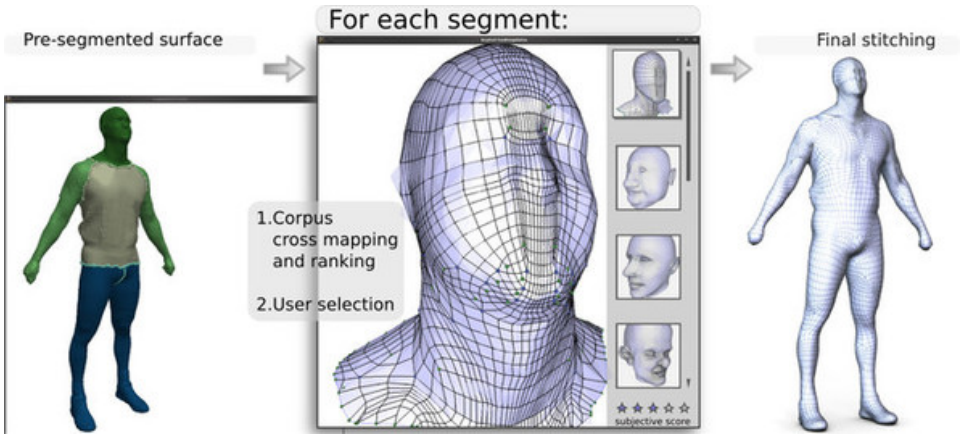


Figure 7.2 – *In addition to regular subdivision, our interactive quadrangulation framework allows the user to manually edit the connectivity applied to each Reeb chart or to select a connectivity among geometrically similar quad-meshes (middle). A final stitching step (right) produces a manifold quad-only mesh.*

ness of the combinatorial Laplacian, yielding a solution with topological guarantees that is exploitable for *certified* geometry processing.

Note however that due to its combinatorial nature, our algorithm will locally break the harmonicity of the function in the vicinity of the removed critical points by flattening the area. This drawback has only a local impact and a small amplitude (in Fig. 7.1(c), $||f - g||_\infty = 0.12\%$) and thus will be acceptable for most applications (like quad-mesh design for example). However, it might be a limitation in specific applications where harmonicity is a critical feature that must be enforced everywhere, like harmonic parameterization for instance.

Nevertheless, such a mechanism has been instrumental in the development of our Reeb-graph based segmentation framework, especially for the manipulation of critical contours and the design of fractional singularities (which correspond to highly degenerate topological configurations). Thanks to this post-process procedure, after each solve of the Laplacian function, spurious critical points were removed with this algorithm, enabling to robustly extract Reeb charts with the appropriate topology.

7.2 Interactive surface quadrangulation

As described in Chapter 5, our Reeb graph based segmentation framework enables to semi-automatically segment a surface into coarse quadrangular charts, with controlled alignment and extraordinary vertices. Since Reeb charts are equipped with local harmonic parameterization, a regular sampling of this parameterization yields quadrangulations of the input

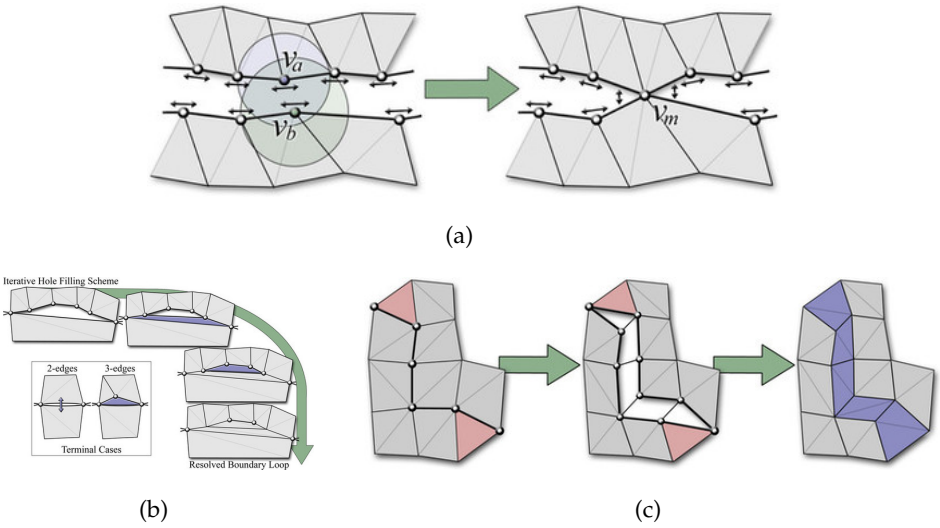


Figure 7.3 – *Three-step stitching procedure. First (a), vertices of adjacent Reeb chart boundaries are greedily matched. Second (b), unmatched vertices are resolved with hole filling procedure. Last (c), possible triangles are resolved in pairs to produce a quad-only output.*

surface. In the following, I describe more advanced interactive algorithms that enable to refine the connectivity of the mesh within each Reeb chart. The connectivity can be manually edited through a set of atomic operations, or automatically transferred from an existing example. With this approach, since adjacent Reeb charts can be edited independently, a stitching procedure is applied in the last step to guarantee a quad-only output, as described in the following.

Stitching procedure

Given two adjacent Reeb charts along with their own quad connectivity, our stitching procedure produces a manifold quad-only output, as described in Figure 7.3. First, vertices are greedily matched on both sides of the boundary based on their Euclidean distance. Second, contiguous segments of unmatched vertices are resolved by inserting series of quads. If such a segment contains an odd number of vertices, triangles are temporarily inserted and later resolved by inserting quads on the shortest path between pairs of triangles, as illustrated in Figure 7.3. Optionally, a last relaxation procedure can be applied to improve the shape of the inserted quads. This procedure enables to robustly stitch the connectivities of each Reeb chart into a manifold quad-only output, while allowing for drastically different meshing strategies on each Reeb chart, as described next.

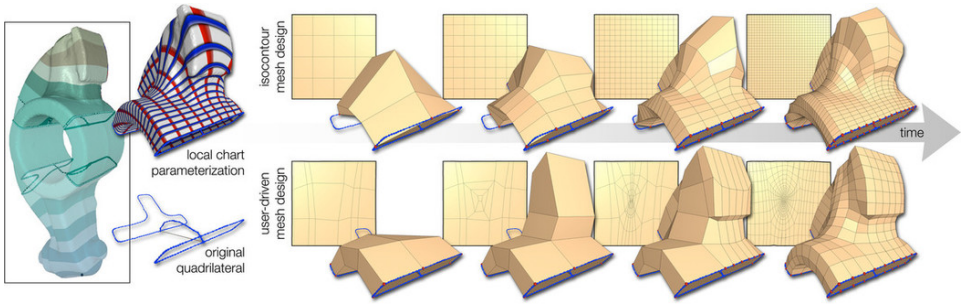


Figure 7.4 – *Connectivity texturing of a challenging geometry (multiple sharp features) on a purposely coarse segmentation. For a given Reeb chart (left), global subdivisions reproduce meshing results from isocontouring (top). In contrast, connectivity texturing maps user designed quadrangulations of the unit square to the parameterized chart (bottom) for improved flexibility and control. Above, we illustrate snapshots of the design process over time (bottom): in this example, the user triggered a few polychord insertions, followed by cube subdivisions to capture the feature corners at the top of the shape, and finally subdivided the texture to obtain the desired sampling density.*

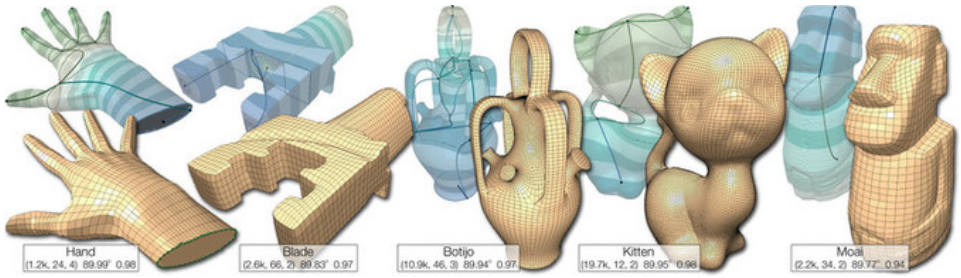


Figure 7.5 – *Examples of user designed quad meshes generated with our editing framework accompanied by quality statistics (vertex count, extraordinary vertex count, max difference from the valence-4, average mesh angle and scaled Jacobian).*

Connectivty texture editing

Given a Reeb chart, we allow the user to interactively define its quad connectivity by designing *connectivity textures*: quad connectivities defined in the planar domain and automatically mapped back to the chart thanks to its local parameterization, as showcased in Figure 7.4. The user is provided with a few atomic operations (vertex movement, polychord insertion/deletion, cube-based subdivision, edge flipping, etc.). Thanks to the local parameterization, each edit is iteratively performed on the surface directly, restricting vertex motions directly to the surface.

Figure 7.5 shows a few examples generated with connectivity texture editing coupled with Reeb-graph based segmentation, along with mesh quality metrics. For all types of edits (Reeb atlas or connectivity texture), our interactive framework responds in less than half second, allowing for truly interactive design sessions. To validate the usability of our system,

Figure 7.6 – *A coarse quadrangulation is extracted from an existing quad-mesh and parameterized (middle). The face is locally improved by connectivity texturing to better capture the nose (right) with no impact on the rest of the mesh (insets).*

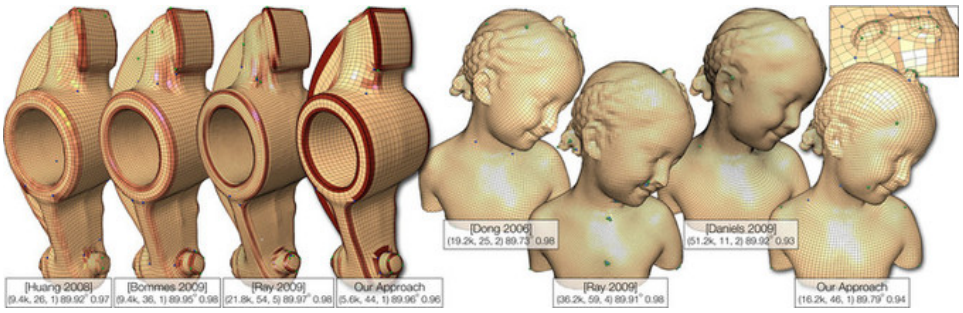


Figure 7.7 – *The comparison of our technique against quad meshes from multiple algorithms illustrates our improved control of extraordinary vertices, i.e. location and valence, as well as element alignment, while producing quality output models for the Rocker Arm and Bimba models. Extraordinary vertices with a valence greater or less than four are respectively reported with blue and green spheres. On the Rocker Arm model, each quadrilateral is color mapped based on the max angle between its normal and its neighbors' normals, from yellow (0°) to dark red (90°). Notice the alignment of the extraordinary vertices and the alignment to sharp features.*

we asked an artist to design a quad mesh for the Bimba model (Figure 7.7, right) with a standard modeling software (Blender, result achieved in 2 hours and 53 minutes) and our approach (result achieved in 31 minutes). From a practical point of view, this experience revealed that the semi-automatic segmentation framework based on the Reeb graph helps users to apprehend surfaces with complex shapes and several prominent features, while connectivity texture editing helps in refining the mesh nearby small-scale details.

Furthermore, our Reeb chart parameterization strategy coupled with connectivity texturing enables to improve existing quad-meshes, by locally refining their connectivity, as illustrated in Figure 7.6.

Figure 7.7 provides further comparisons with results obtained with our approach and more traditional, automatic techniques (HZM*08, BZK09, RVAL09). These results show that meshes with compatible quality metrics (the same as shown in Figure 7.5) can be obtained with our approach but with a better feature alignment and extraordinary vertex control.

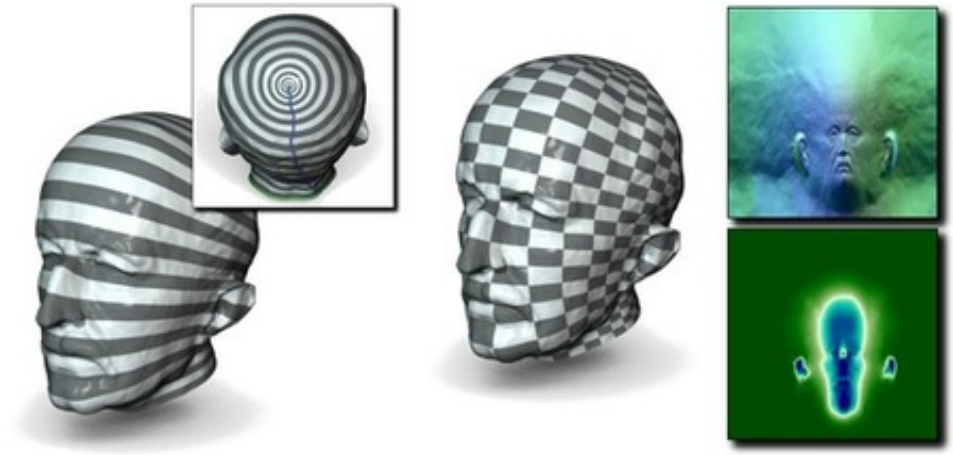


Figure 7.8 – *Geometry unfolding procedure. From left to right: initial harmonic field (inset: pole and streamline constraints), Reeb chart parameterization, unfolded normal map (top) and unfolded conformal factor (bottom). This generic unfolding procedure enables to reduce the problem of geometry aware cross parameterization to that of picture registration.*

Connectivity texture transfer

In addition to allow for an interactive editing of the connectivity textures, we also developed an approach for their transfer from existing quad meshes. This approach relies on a fast algorithm for the automatic cross parameterization of surfaces of disc or annulus topology.

Given two Reeb charts $\mathcal{S}_1$ and $\mathcal{S}_2$ with disc topology (a similar strategy is derived for Reeb charts with annulus topology), each of them can be unfolded to the plane (as illustrated in Figure 7.8) with the strategy described in Chapter 5, yielding two planar maps $\phi_1 : \mathcal{S}_1 \rightarrow \mathcal{D}_1 \subset \mathbb{R}^2$ and $\phi_2 : \mathcal{S}_2 \rightarrow \mathcal{D}_2 \subset \mathbb{R}^2$. Each of these will yield some area distortion $\lambda_1 : \mathcal{D}_1 \rightarrow \mathbb{R}^+$ and $\lambda_2 : \mathcal{D}_2 \rightarrow \mathbb{R}^+$, denoting the stretch one needs to apply to the surface to unfold it to the plane, as illustrated in Figure 7.8 (right, bottom). By constraining these parameterization to map to a unit square, our strategy reduces the problem of cross parameterization to that of picture registration, by considering the picture of the area distortion, that we call *quasiconformal encoding*.

Then, to compute a cross map of low-distortion between $\mathcal{S}_1$ and $\mathcal{S}_2$, one needs to find an optimal planar map $\hat{\psi}^* : \mathcal{D}_1 \rightarrow \mathcal{D}_2$. The corresponding 3D bijection $\psi^* : \mathcal{S}_1 \rightarrow \mathcal{S}_2$ can be then directly retrieved by considering $\psi^* = \phi_2^{-1} \circ \hat{\psi}^* \circ \phi_1$.

In particular, to achieve low-distortion bijections, we consider the fol-

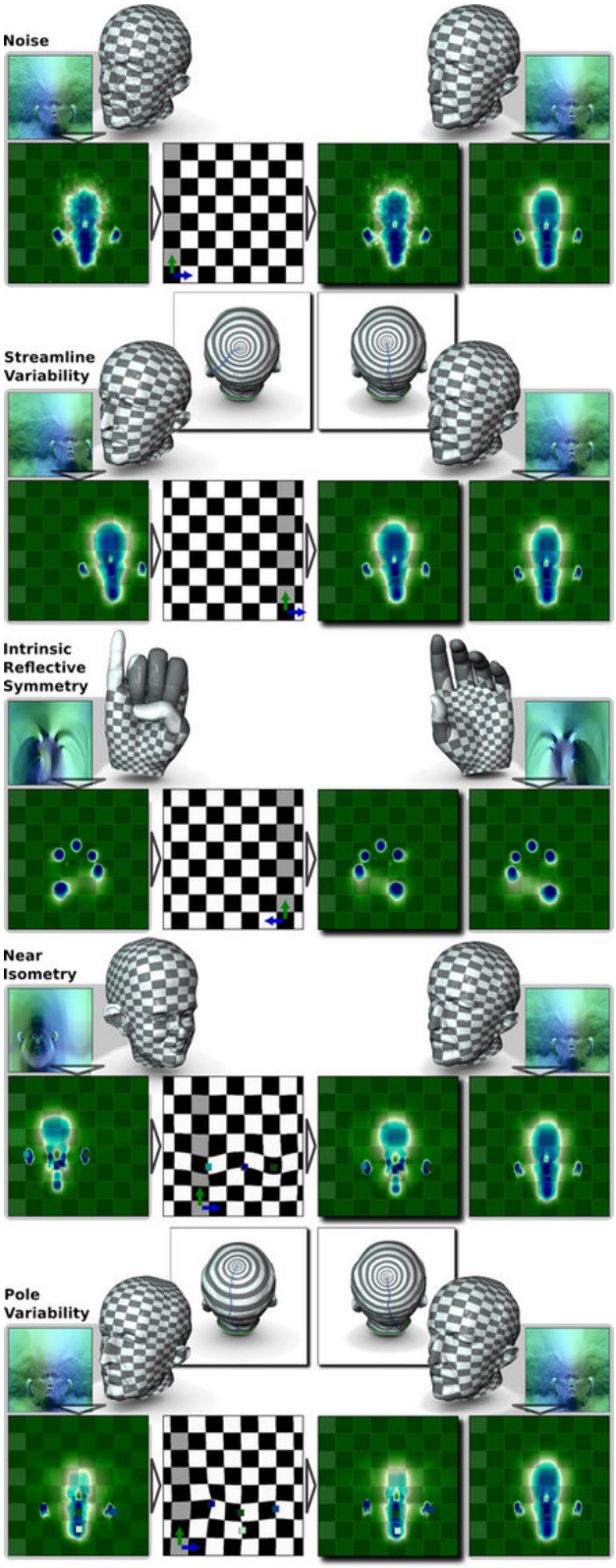


Figure 7.9 – *Variability of the quasiconformal encoding (from top to bottom): random noise can be registered with the identity, streamline variability (pure isometry) translates into periodic translation, intrinsic reflective symmetry translates into axial symmetry, near isometries and pole variability translate into as-rigid-as possible planar maps. In the center of each row, the optimal canonical cross map found automatically by the solver.*

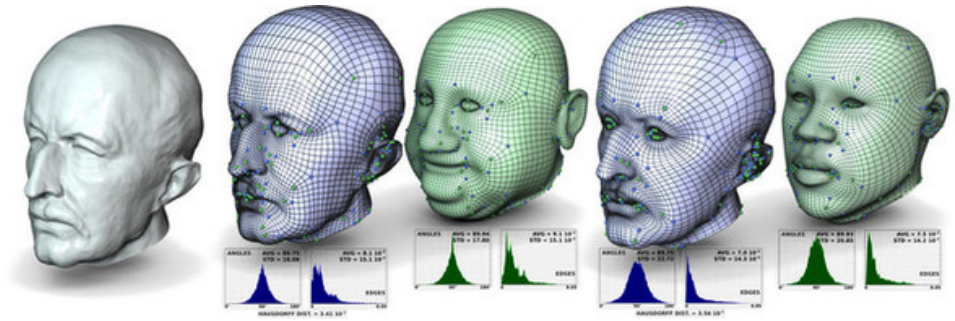


Figure 7.10 – *Examples of connectivity transfer from manually designed quad-meshes (green) to a scanned geometry (blue).*

lowing functional for minimization:

$$E(\hat{\psi}) = \int_{(u,v)\in\mathcal{D}_1} (\lambda_1(u,v) - \lambda_2(\hat{\psi}(u,v)))^2 du dv \tag{7.3}$$

Intuitively, a minimized energy will correspond to a planar cross map that maps regions of similar area distortion (and consequently of similar shapes in 3D).

To restrict the space of candidate planar maps $\hat{\psi}$, we study the variability of our quasiconformal encoding. As illustrated in Figure 7.9, the variability in the placement of the initial streamline for subsequent parameterization can be compensated by a periodic rotation of the quasiconformal encoding (second row). Next, intrinsic reflective symmetry can be compensated by an axial symmetry of the quasiconformal encoding (third row). Finally, near isometries can be compensated with low distortion by extracting the most persistent maxima of λ and using them as constraints for as-rigid-as-possible planar maps (SMW06) (fourth row). Consequently, we consider as candidate maps for our optimization procedure compositions of periodic translations, axial symmetries and as-rigid-as-possible planar maps. Since the latter maps can be expressed with an analytic expression, we derived an analytic expression of their gradient, allowing to formulate the gradient of our functional E (Equation 7.3) and to use it in a fast, parallel, multi-grid gradient descent optimization solver (which took at most 10 seconds to compute in our experiments).

Figure 7.10 shows typical examples of connectivity transfers obtained automatically with our algorithm, by cross-mapping manually designed quad-meshes onto a scanned geometry. In particular, note that the extraordinary vertex layout is nicely preserved and adjusted to the input geometry. Figure 7.11 shows additional results of local connectivity transfers and final stitching to produce a quad-only manifold output.

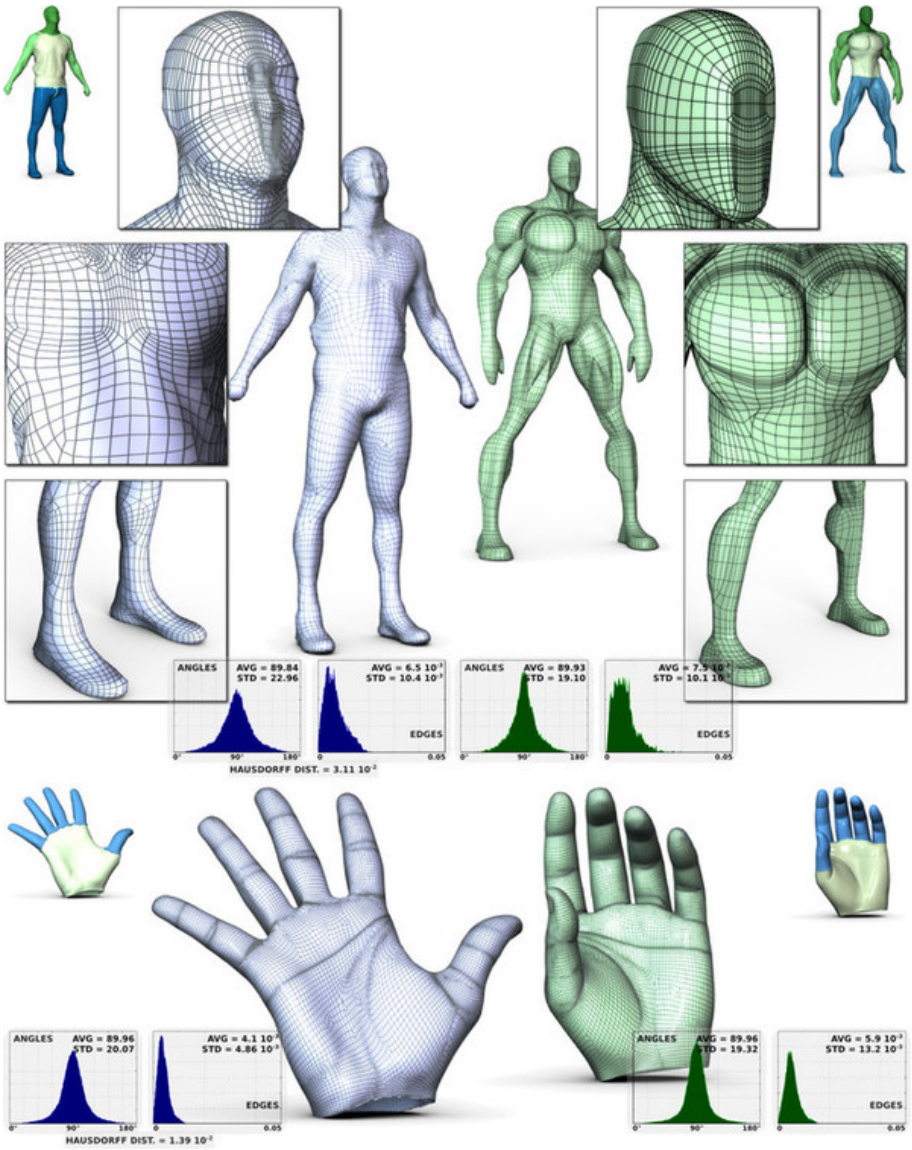


Figure 7.11 – *Examples of connectivity transfer and stitching between designed quad meshes (green) to scanned geometries (blue).*

Concluding remarks

In this section, I extended the description of our Reeb-graph based framework for the interactive design of surface quadrangulation. With the notion of connectivity texture editing, we provided users with intuitive advanced control for the capture of small scale details. With the notion of connectivity texture transfer, we enabled users to easily re-use previously designed connectivity textures. Combining these two algorithms will hopefully improve the productivity of modeling artists by helping them rapidly prototype quad-meshes by extending and adapting through connectivity texture editing already existing meshes.

7.3 Panorama stitching

The composition of panoramas from a collection of smaller individual images has recently become a popular application in digital image processing. The composition process is composed of 3 steps, usually executed independently one after the other: *(i)* image registration, *(ii)* seam computation and *(iii)* color correction.

The first step *(i)* registers all the images onto a common domain by optimizing with Bundle Adjustment the transformation of each image relatively to its neighbors, found through automatic feature point matching. The second step *(ii)* automatically determines which pixels should be shown in the final composition for regions where multiple registered images overlap. The last step *(iii)* aims at correcting large color variations across image boundaries due to exposure difference and is usually achieved by solving a Poisson system on the image gradient.

The second step (seam computation) can be expressed as a minimum cut problem and is traditionally executed with time-consuming solvers, such as Graph Cuts (BK04), making this step of the pipeline highly computationally demanding and not conducive to user interactions.

In this work, we address this problem by introducing a new algorithm, called *Panorama Weaving*, for the fast automatic seam creation and interactive editing. Interestingly, as described further down, the core of the algorithm relies on an analysis of the topology of the layout of registered images.

Optimal seams

Given a collection of n panorama images $I_1, I_2, \ldots, I_n$ and the panorama P, the optimal seam computation problem can be thought of as finding a discrete labeling $L(p) \in (1...n)$ for all panorama pixels $p \in P$ which minimizes the transition between each image, in order to minimize visual artifacts in the subsequent color correction. If $L(p) = k$, this indicates that the pixel value for location p in the panorama comes from image I_k. Boundaries between regions of the panorama having a different label $L(p)$ are called *seams*.

These transitions can be defined by an energy on the smoothness $E_s(p, q)$ of the labeling of neighboring elements $p, q \in N$, where N is the set of all neighboring pixels. We would like to minimize the sum of the energy of all neighbors, E. For the optimal seam computation problem,

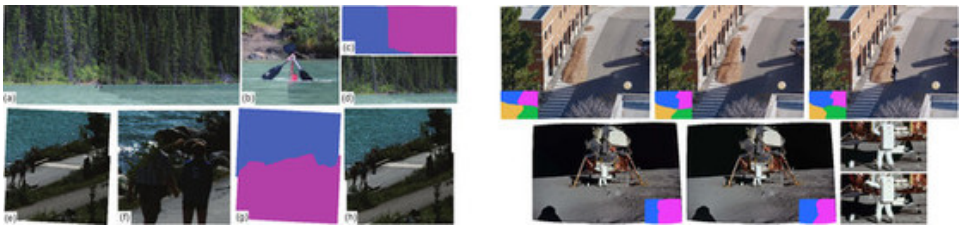


Figure 7.12 – *Techniques for automatic seam creation (such as Graph Cuts) may produce results with non-desirable, yet exactly optimal seams (left). Even when seams are visually acceptable for the user, moving elements in the scene may cause multiple visually valid seam configurations (right).*

this energy is typically defined as:

$$E(L) = \sum_{p,q \in N} E_s(p,q) \tag{7.4}$$

Several energy functionals E_s can be considered for this problem. For instance, to minimize the transition in pixel values, the following energy (penalizing pixel variations across the seams) can be considered:

$$E_s(p,q) = ||I_{L(p)}(p) - I_{L(q)}(p)|| + ||I_{L(p)}(q) - I_{L(q)}(q)|| \tag{7.5}$$

Penalizing gradient variations across the seams can also be considered:

$$E_s(p,q) = ||\nabla I_{L(p)}(p) - \nabla I_{L(q)}(p)|| + ||\nabla I_{L(p)}(q) - \nabla I_{L(q)}(q)|| \tag{7.6}$$

Note that $L(p) = L(q)$ implies $E_s(p,q) = 0$. Therefore, only pixel neighbors separated by a seam intervene in the energy evaluation.

The optimization of this functional (called the *min-cut* problem) can be achieved with algorithms such as Graph Cuts, which is probably the most popular technique so far for seam creation. However, as discussed before, this algorithm is computationally expensive. Also, at a technical level, it is not guaranteed to find a global minimum of the functional for multiple labeling (if multiple images overlap in the panorama). More importantly, from a practical point of view, even numerically optimal seams may not produce visually pleasing results, as illustrated in Figure 7.12 (left) where only half of a person in shown in the final panorama (bottom). Also, even if the seams can be acceptable for the user (Figure 7.12, right), several alternative valid seam configurations are possible.

In this work, we address this issue by introducing a fast seam creation technique that enables user edits at interactive rates.

Min-cut min-path duality

In the specific case of a binary labeling (regions where only two images overlap, Figure 7.13 left), it has been shown that the min-cut problem is

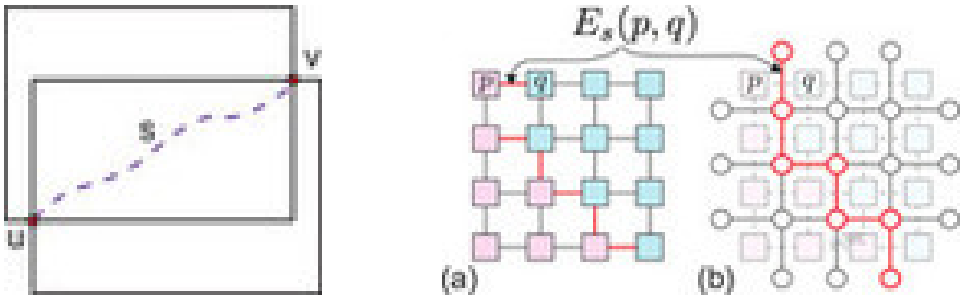


Figure 7.13 – *Min cut-labeling duality on a two-image arrangement (left). The 4-neighborhood min-cut solution (right, a) with its dual min-path solution (right, b). The min-cut labeling is colored in red/blue and the min-path solution is highlighted in red.*

equivalent to the min-path problem on the dual graph (Has81). In practice, for panoramas, this corresponds to a graph where each vertex represents a four-neighborhood of pixels and an edge connects adjacent four-neighborhoods sharing two pixels, and is associated with the metric E_s (Figure 7.13, right). Given this metric, such a length minimizing path can be computed by evaluating the distance field from a corner of the image intersections (u in Figure 7.13, left) and computing the backward integral line from the other corner (v in Figure 7.13, left). Therefore, in the case of pairwise seams, it is possible to compute the optimal seam very efficiently: the distance field evaluation takes $O(nlog(n))$ steps (where n is the number of pixels in the image overlap) and the integral line is computed in linear time ($O(n)$ steps).

Moreover, this setting is very conducive to user interactions. Given two corners of image overlaps (u and v in Figure 7.13, left), one can actually compute two distance fields in parallel: one (noted d_u) having u as origin and another one (noted d_v) having v as origin. Next, given a point w of the overlap prescribed by the user, the optimal seam passing through w can be computed in linear time by computing backward integral lines from w on d_u and d_v (terminating at u and v). Once this is accomplished, a new distance field (from w) can be computed in parallel in the background to support further constraint insertion. In practice, this strategy enables to explore the space of possible optimal seams given sparse user constraints at truly interactive rates (with only linear passes with the size of the overlap).

Therefore, in the case of pairwise seams (binary labels), one can efficiently compute and edit optimal energy seams. In the following, I describe how to generalize this approach to multiple labels (panorama made of multiple images).

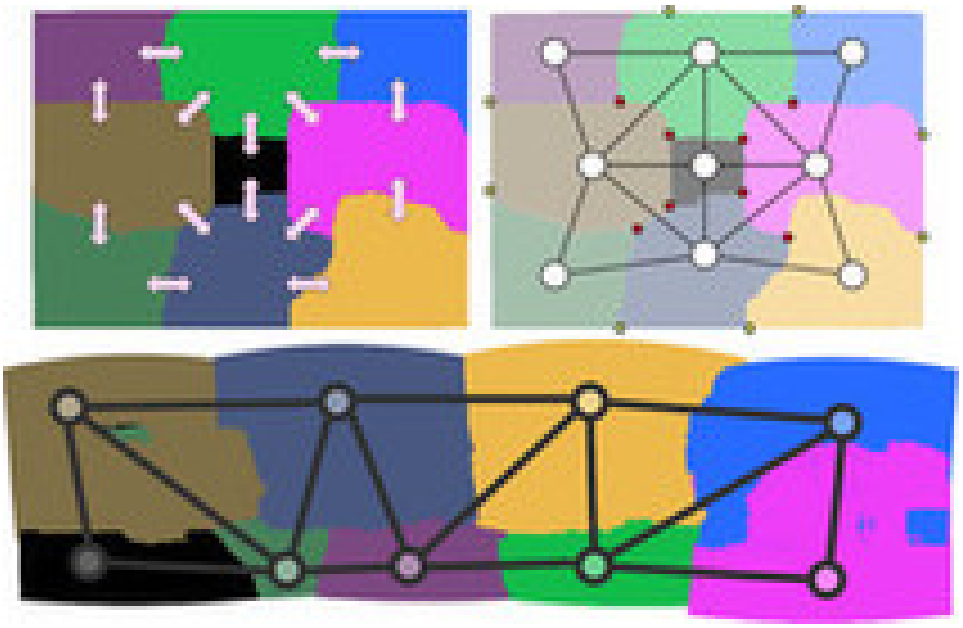


Figure 7.14 – *A solution to the panorama boundary problem (top left) can be considered as a network of pairwise boundaries between images (top right). Our adjacency mesh representation is designed with this property in mind. Nodes correspond to panorama images, edges correspond to boundaries and branching points (intersections in red) of pairwise seams correspond to faces of the mesh. Graph Cuts optimization can provide more complex pixel assignments (bottom) where "islands" of pixels assigned to one image can be completely bounded by another image. Our approach simplifies the solution by removing such islands.*

Seam network topology

To generalize the approach described in the previous paragraph to panoramas made of multiple images, we aim at constructing a proper collection of pairwise seams, called a *seam network*. This construction is based on the observation, illustrated in Figure 7.14 (top left), that a label assignment mostly forms a simple collection of regions separated by pairwise seams (double arrows in Figure 7.14, top left). Moreover, these pairwise seams meet in specific configurations (called *k-branching points*, red circles in Figure 7.14, top right), where all the k images involved in the meeting pairwise seams overlap. One can depict the overall topology of such a seam network by considering the dual notion of *Adjacency Mesh* (Figure 7.14, top right), where each vertex represents a region of unique label assignment (white circles), each edge represents a pairwise seam (black edges) and each k-sided polygon represents a k-branching point. Given an adjacency mesh, the corresponding seam network can be easily computed by *(i)* locating the branching point corresponding to each polygon

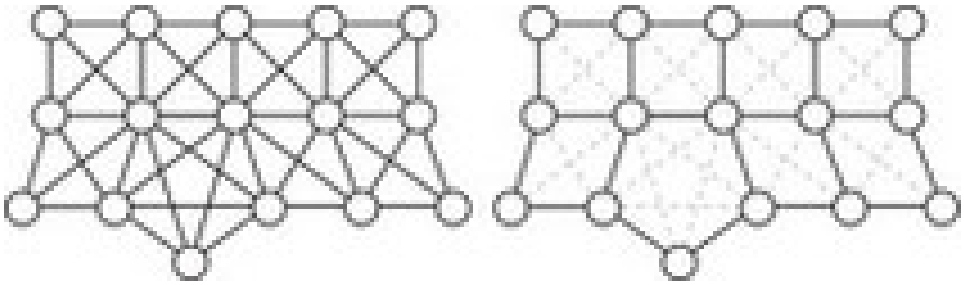


Figure 7.15 – *Considering the full neighborhood graph of a panorama (left), where an edge exists if an overlap exists between a pair of images, an initial valid adjacency mesh (right) can be computed by finding all non-overlapping, maximal cliques in the full graph then activating and deactivating edges based on the boundary of each clique.*

(described further down) and *(ii)* computing the pairwise seam between them (corresponding to each edge of the adjacency mesh).

Therefore, to generalize pairwise seams to panoramas made of multiple images, one needs to derive an algorithm for the automatic computation of the adjacency mesh of the panorama. In the following, given a registered set of images, we only assume that [1]:

- Each image contributes to a unique simply-connected component of the final panorama;
- A set of k images where every pair of images overlap contains at least one pixel where all k images overlap.

Given a set of registered images, the full neighborhood graph is first constructed (Figure 7.15, left). Each vertex of this graph represents a registered image and an edge connects two vertices if there exists an overlap between their registered images. The faces of the adjacency mesh correspond to configurations where k images overlap, allowing for the junction of k pairwise seams into a k-branching point. Therefore, the faces are constructed by extracting the maximal cliques in the full neighborhood graph. Next, the edges of the adjacency mesh are selected as edges of the full neighborhood graph shared by two faces. Finally, the vertices of the adjacency mesh are given by these of the full neighborhood graph.

Once the adjacency mesh is computed, branching points need to be extracted. As illustrated in Figure 7.16, the k boundary overlap-corners of all the images involved in the face (yellow circles, left) are used as origin for distance field computations (right). The k branching point is then chosen as a heuristic as the minimizer of the sum of these k distance fields. Next,

[1] A follow-up work (STB*13) describes a generalization of this algorithm that no longer needs these requirements and that handles arbitrary configurations.

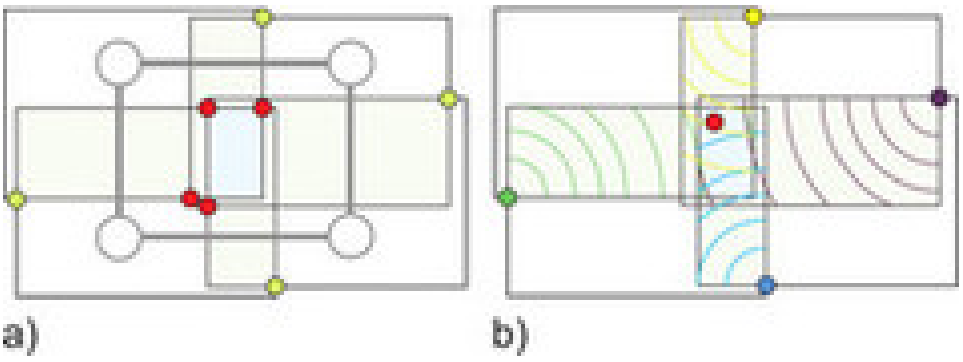


Figure 7.16 – *Pairwise seam endpoints closest to a multi-overlap (a, red) are considered a branching point. This can be determined by finding a minimum point in the multi-overlap with respect to min-path distance from the partner endpoints (b).*

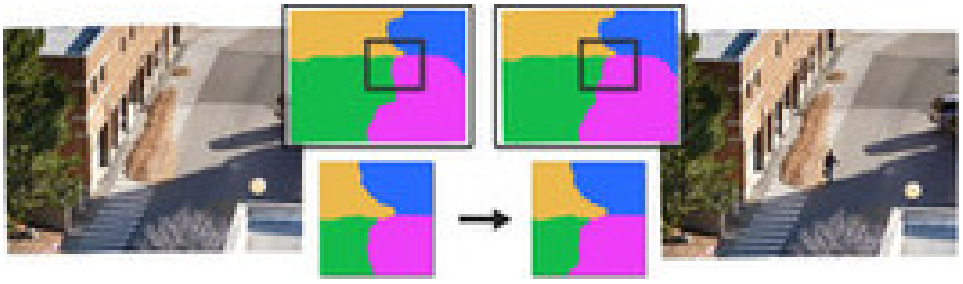


Figure 7.17 – *Importing a seam network from another algorithm. The user is allowed to import the result generated by Graph Cuts (left) and adjust the seam between the green and purple regions to unmask a moving person (right). Note that this edit has only a local effect, and that the rest of the imported network is unaltered.*

the pairwise seams for each mesh edge are computed (linking branching points together or branching points to overlap corners on the boundary of the panorama). Note that branching point computation as well as pairwise seam computation are local operations. Therefore, branching points are first computed in parallel. Second, pairwise seams are computed in parallel. Finally, since pairwise seams are computed independently, they may self-intersect. Such configurations are easily resolved by collapsing pairs of paths from the intersection to the next branching point.

At this point, the seam network is fully computed and is ready for interactions as described in the previous paragraph dedicated to pairwise seams. Moreover, the location of a branching point can be edited with a similar mechanism while its valence can be edited too by splitting it into smaller valence branching points (which is easily achieved by subdividing the corresponding k-sided polygon in the adjacency mesh).

Also note that our interactive algorithm can also be used to edit the output of other labeling algorithms such as Graph Cuts, as illustrated in Figure 7.17. Given a labeling of the pixels of the panorama, the al-

Data-set	MP	Images	PW-P (s.)	PW-S (s.)	GC-S	E. Ratio
Crosswalk	16.7	4	1.3	7.2	369.6	0.995
Fall-5way	30.0	5	2.4	12.1	735.4	1.220
Skating	44.7	6	3.2	16.8	734.0	0.851
Lake	9.4	22	0.5	2.9	337.2	0.503
Graffiti	36.6	10	4.3	19.6	983.7	0.707
Nation	49.1	9	4.6	23.2	1168.7	0.800

Table 7.1 – *Performance results comparing Panorama Weaving to Graph Cuts for our datasets. Panorama Weaving run serially (PW-S) computes solutions quickly. When run in parallel, runtimes are reduced to just a few seconds. The energy ratio (E. ratio) between the final seam energy produced by Panorama Weaving and Graph Cuts (PWEnergy / GC Energy) is shown. For all but one dataset, Weaving produces a lower energy result and is comparable otherwise. Panorama image sizes are reported in megapixels (MP).*

gorithm first extracts the boundaries of the regions. Then, branching points (boundary intersections) are extracted. Next, each boundary segment (bounded by two branching points) is identified as a seam and the connected components of the resulting seam network are identified. To be compatible with our framework, only the seam networks made of a single connected component can be imported. Thus we only consider the biggest connected component of the network and small islands are discarded. Finally, our seam data-structures are fed with the seam network and the adjacency mesh is updated if necessary. Since the editing operations do not cascade globally, a user can edit a problem area locally and maintain much of the original solution if desired.

Results

In this section, I detail the results in both the creation and editing phases of our system. All results were obtained on a 3.07 GHz Intel i7 4-core processor (with Hyperthreading) with 24 GB of memory.

Table 7.1 provides a performance comparison between our approach and the state-of-the-art technique, Graph Cuts (implementation provided by the authors, which many consider as the exemplary implementation). Not having an equally well-established in-core parallel implementation for Graph Cuts, we first use a serial version of our algorithm (PW-S) for comparison, yielding speedups of up to two orders of magnitude. Such speedups are further amplified with the parallel implementation of our algorithm (PW-P), which shows nearly ideal parallel speedup. More surprisingly, our approach, which can be seen as a heuristic with regard to

Figure 7.18 – *Panorama Weaving on a challenging data-set (Nation, 12848 x 3821, 9 images) with moving objects during acquisition, registration issues and varying exposure. Our initial automatic solution (bottom, left) was computed in 4.6 seconds at full resolution for a result with lower seam energy than Graph Cuts. Additionally, our interactive system for the user exploration of the seam solution space (bottom, right) easily enables: (a) the resolution of moving objects, (b) the hiding of registration artifacts (split pole) in low contrast areas (scooter) or (c) the fix of semantic notions for which automatic decisions can be unsatisfactory (stoplight colors are inconsistent after the automatic solve). The user editing session took only a few minutes. (top) the final, color-corrected panorama.*

the min cut problem, produces for all but one data-set (Fall-5way) better energy seams. This can be explained by the fact that this data-set has originally a high valence branching point and such a topological constraint penalizes the resulting energy (note that this branching point can be interactively split, resulting in lower energy). For the other examples, the improvement over Graph Cuts can be explained by the fact that our approach does not produce isolated islands of label assignments by construction, while these often occur with Graph Cuts (Figure 7.15, bottom), increasing the resulting energy.

As discussed previously, our approach enables to interactively explore the space of seam networks with interactive rate responds, given user defined sparse constraints. In particular, the user is offered the possibility to insert point constraints on seams to bend them over scene objects, to move branching points and to edit their valence. These capabilities are illustrated in Figure 7.18, which shows a challenging data-set for automatic techniques, due to registration artifacts, moving objects during acquisition and exposure variations. As detailed in the lateral in-sets, point constraint and branching point movement enables to bend seams around moving objects (a), to hide registration artifacts (b) or to fix semantic notions for which automatic decisions can be unsatisfactory (c).

Concluding remarks

In this section, I described an efficient technique for pairwise seam computation and editing for panorama creation. In particular, I showed how this approach could be generalized to panoramas made of multiple images by reasoning on the topology of the layout of registered images. Not only this approach is orders of magnitude faster than the state-of-art, but it also produces results of higher quality (lower residual energy). In addition to these contributions for the automatic seam computation, this approach also provides to the best of our knowledge the first effective machinery for interactive seam editing. We therefore consider this algorithm as the reference of the problem of seam creation and editing in panoramas. These contributions have been instrumental in the development of follow-up techniques, such as on-the-fly panorama acquisition and creation (STB*13), seamless composition (SGSP15), or large-scale panorama creation (PST*15).

7.4 Visualisation verification

Scientific Visualization has become a standard component in scientific software. However, unlike traditional components of the scientific pipeline (such as mathematical modeling or numerical simulation), only few research efforts have been devoted to the verification of the accuracy, reliability and robustness of its algorithms. As discussed in Chapter 3, isosurfaces are fundamental geometrical objects for the analysis and visualization of scalar data. While an approach has been described to evaluate the geometrical accuracy of isosurface codes for (ESN*09), no verification framework was available to assess their topological accuracy. This problem is addressed by this work.

As described in Chapter 3, isosurface extraction on PL scalar fields defined on PL manifolds can be achieved robustly thanks to the properties of the linear interpolant with barycentric coordinates. However, when considering regular grids, it is often desirable in practice not to subdivide them into PL manifolds, to reduce the memory footprint. Regarding piecewise trilinear (PT) scalar fields defined on regular grids, the possible presence of critical points in the interior of the cells challenges, from a topological point of view, the robust extraction of isosurfaces with algorithms derived from the celebrated *"Marching Cubes"* algorithm (LC87).

In this work, we introduced a verification framework composed of sim-

ple and effective algorithms for the topological verification of isosurfacing codes processing PT scalar fields defined on regular grids. Note that the simplicity of a verification algorithm is of major importance since a simple implementation is less likely to contain bugs. In particular, given a manufactured solution (a PT scalar field whose topology can be easily and robustly evaluated by our algorithms), our framework is composed of three steps:

1. *Consistency check:* verifying that the output produced by an isosurfacing code is consistent;
2. *Betti number verification:* for closed isosurfaces, verifying that the output produced by an isosurfacing code is homemorphic to the manufactured level-set;
3. *Euler characteristic:* for isosurfaces with boundary, verifying that the output produced by an isosurfacing code has the same Euler characteristic as the manufactured level-set.

Consistency check

Level sets of PT scalar fields for non-critical values are piecewise trilinear 2-manifolds that are approximated in practice by isosurfacing codes as PL 2-manifolds. Therefore, our framework starts by assessing the *consistency* of an isosurfacing code by checking if its output is indeed manifold. This is achieved by checking if the link of any interior vertex is indeed a closed, connected PL 1-manifold and that the link of any boundary vertex is indeed an open, connected PL 1-manifold.

In order to stress the consistency test we generate a random scalar field with values in the interval $[-1, 1]$ and extract the isosurface with isovalue $\alpha = 0$ using a given isosurfacing technique, subjecting the resulting triangle mesh to the consistency verification. This process is repeated a large number of times. If the implementation fails to produce PL-manifolds for all cases, then the counterexamples provide documented starting points for debugging. If it passes the tests, we consider the implementation verified.

Betti number verification for closed isosurfaces

Next, in order to verify that the output surface has indeed the correct topology, we check if it is homemorphic to a manufactured isosurface by

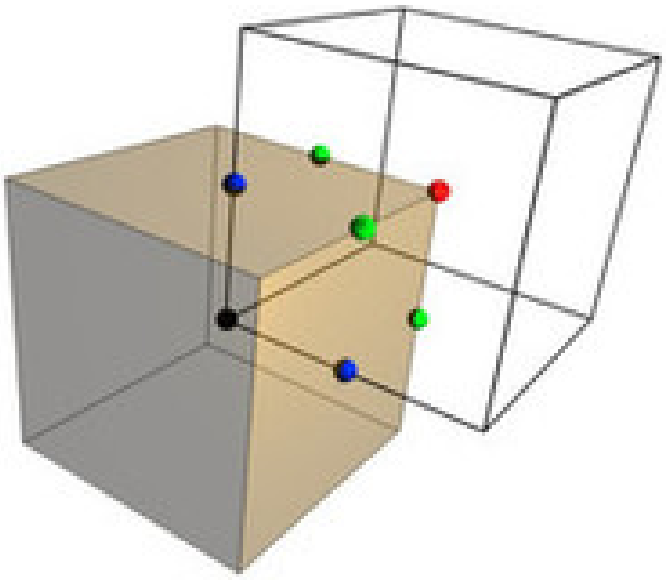


Figure 7.19 – *The four distinct groups of vertices O, F, E, C (original, face, edge, center) are depicted as red, blue, green and black spheres (semi-transparent cube: voxel of $\mathcal{G}$).*

comparing their Betti numbers. In particular, random scalar fields with values in the interval $[-1, 1]$ are generated and the isosurface with isovalue $\alpha = 0$ is considered. In the following, only closed isosurfaces will be considered, therefore isosurfaces with boundary are avoided by assigning the scalar value 1 to every vertex on the boundary of the regular grid. To robustly evaluate the Betti numbers of that isosurface, we introduce a simple algorithm derived from Digital Topology.

Let $f : \mathcal{G} \to \mathbb{R}$ be a PT scalar field defined on a $n \times n \times n$ cubic regular grid. Consider now the $2n \times 2n \times 2n$ regular grid $\mathcal{G}'$ obtained by refining $\mathcal{G}$ and $f' : \mathcal{G}' \to \mathbb{R}$ given by the trilinear interpolation of f (therefore, f and f' are identical for each point of $\mathcal{G}'$). As illustrated in Figure 7.19, the vertices of $\mathcal{G}'$ can be classified into four distinct sets, denoted O, F, E, C. The set O contains the vertices of $\mathcal{G}'$ that are also vertices of $\mathcal{G}$. F, E and C contain the vertices of $\mathcal{G}'$ lying on the center of faces, edges and voxels of $\mathcal{G}$ respectively.

Given a scalar value α, the *digital object* $\mathcal{O}_\alpha$ is the subset of voxels $v \in \widehat{\mathcal{G}'}$ (where $\widehat{\mathcal{G}'}$ is the dual regular grid of $\mathcal{G}'$, for which each voxel corresponds to a vertex of $\mathcal{G}'$) such that at least one of the criteria below is satisfied:

1. $v \in O$ and $f(v) \leq \alpha$
2. $v \in F$ and both neighbors of v in O have scalars less than (or equal to) α

3. $v \in E$ and at least 4 of 8 neighbors of v in $O \cup F$ have scalars less than (or equal to) α

4. $v \in C$ and at least 12 of the 26 neighbors of v in $O \cup F \cup E$ have scalars less than (or equal to) α

The description above is called *Majority Interpolation* (MI) and allows to compute the voxels that belong to a digital object $\mathcal{O}_\alpha$.

The importance of $\mathcal{O}_\alpha$ is three-fold. First, the boundary surface of the union of the voxels in $\mathcal{O}_\alpha$, denoted $\partial\mathcal{O}_\alpha$ and called a *digital surface*, has been shown to be a 2-manifold (SLS07). Second, the genus of $\partial\mathcal{O}_\alpha$ can be computed directly with a simple algorithm (CR08). As the connected components of $\mathcal{O}_\alpha$ can also be easily computed and isolated, one can calculate the Euler characteristic χ of each connected component of $\mathcal{O}_\alpha$ and thus the Betti numbers β_0, β_1 and β_2 for $\partial\mathcal{O}_\alpha$. Third, given a level set $f^{-1}(\alpha)$ without boundary of a PT scalar field $f : \mathcal{G} \to \mathbb{R}$, $\partial\mathcal{O}_\alpha$ is homemorphic to $f^{-1}(\alpha)$ if $\mathcal{G}$ contains no ambiguous voxel (a voxel is not ambiguous if all positive vertices form a single connected component via the positive edges and all negative vertices form a single connected component via the negative edges; otherwise it is ambiguous (vGW94), see (ENS*12) for a proof). Therefore, given a PT scalar field inducing no ambiguous voxel, our algorithm computes in a robust and simple manner the Betti numbers of any closed isosurfaces.

In practice, given a random scalar field, $\mathcal{G}$ is refined multiple times, until it no longer exhibits ambiguous voxels. Next the Betti numbers of the manufactured isosurface $f^{-1}(\alpha)$ is evaluated with our algorithm. Finally, the Euler characteristic of the surface produced by the isosurfacing code under verification is evaluated (Equation 3) and the Betti numbers are deduced from it and compared with those returned by our algorithm. If both Betti number evaluations match, the verification succeeds.

Euler characteristic verification for open isosurfaces

In order to cover the case of open isosurfaces, we add a third test in our verification framework that compares the Euler characteristic of the surface returned by the isosurfacing code under verification with that of a manufactured isosurface.

In the smooth setting, it is known from Morse theory (Mil63) that the Euler characteristic of a level set changes in the vicinity of a critical point p. In particular, if ϵ_p is a small neighborhood around p and $L^-(p)$ and $L^+(p)$

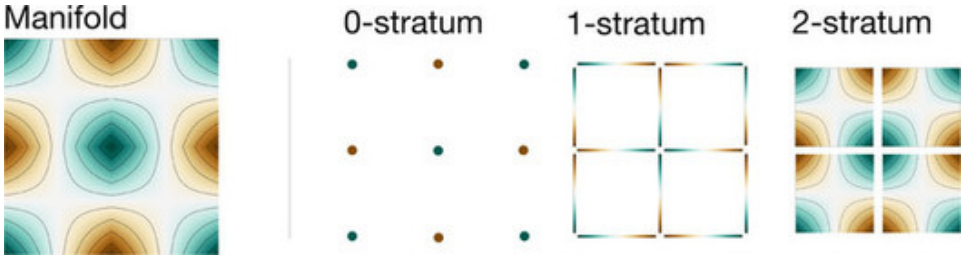


Figure 7.20 – *Illustration of the stratification of a manifold. The colormap illustrates the value of each point of the scalar field. Although the field is not everywhere differentiable, its restriction to each stratum is (exept for 0-strata).*

are the subset of the points of $\partial\epsilon_p$ satisfying $f(x) < f(p)$ and $f(x) > f(p)$ respectively, then the change in the Euler characteristic, denoted by $\Delta\chi(p)$, can be expressed as:

$$\Delta\chi(p) = \chi(L^+(p)) - \chi(L^-(p)) \tag{7.7}$$

In the following, we extend this reasoning to easily evaluate the Euler characteristic of a manufactured isosurface, with a robust algorithm for PT scalar fields. Given a random scalar field f taking values in the interval $[-1, 1]$, cells containing critical points are identified based on their configuration of *positive* and *negative* vertices. Next, for each of these, the location and value of the critical point is numerically estimated. In particular, given the trilinear interpolant, f can be expressed as: $f(x, y, z) = axyz + bxy + cxz + dyz + ex + fy + gz + h$ and the location p of a critical point is given by:

$$\begin{aligned} p_x &= \frac{d\Delta_x \pm \sqrt{\Delta_x\Delta_y\Delta_z}}{a\Delta_x} \\ p_y &= \frac{c\Delta_y \pm \sqrt{\Delta_x\Delta_y\Delta_z}}{a\Delta_y} \\ p_z &= \frac{b\Delta_z \pm \sqrt{\Delta_x\Delta_y\Delta_z}}{a\Delta_z} \end{aligned} \tag{7.8}$$

with $\Delta_x = bc - ae$, $\Delta_y = bd - af$, and $\Delta_z = cd - ag$. Critical points on faces of voxels are found by setting x, y or z to either 0 or 1, and solving the linear equations. The scalar field is regenerated if any degenerate critical point is detected. Additionally, to avoid numerical instability, it is also regenerated if any interior critical point lies too close to the border of a cell. Next the Euler characteristic of a manufactured isosurface is evaluated with the following approach.

To extend smooth Morse theory to piecewise smooth scalar fields (such as PT scalar fields), we consider Stratified Morse theory. Intuitively, a stratification is a partition of a piecewise-smooth manifold such that each

subset, called a stratum, is either a set of discrete points or has smooth structure. In the case of regular grids, the stratification we propose will be formed by four sets (the strata), each one a (possibly disconnected) manifold. The vertex set contains all vertices of the grid. The edge set contains all edge interiors, the face set contains all face interiors, and the cell set contains all cube interiors, as illustrated in Figure 7.20. The important property of the strata is that the restriction of f to each stratum is differentiable (or lack any differential structure, in the case of the vertex-set). In this setting, one can apply standard Morse theory on each stratum, and then combines the partial results appropriately.

To determine the Euler characteristic of a manufactured isosurface at isovalue α, we proceed as follows:

1. Critical points and critical values are identified for each 3-dimensional and 2-dimensional stratum as described previously (Equations 7.8). In addition, every point in the vertex set is considered as critical as well (not all critical points induce topological changes in level sets in Stratified Morse theory).

2. For each critical point p identified previously, $\chi(L^-(p))$ and $\chi(L^+(p))$ are evaluated. Given a critical point p of a d-dimensional stratum, let $T_\epsilon(p)$ be a small neighborhood in the stratum. The *lower tangential link* $T_{L^-}(p)$ is the set of points of $T_\epsilon(p)$ with lower f values than p. Similarly, let $N_\epsilon(p)$ be a small neighborhood of the $(3-d)$-dimensional submanifold that is transverse to the stratum and going through p. The *lower normal* $N_{L^-}(p)$ is the set of points on the boundary of $N_\epsilon(p)$ with lower f values than p. Then, $\chi(L^-(p)) = \chi(N_{L^-}(p)) + \chi(T_{L^-}(p)) - \chi(N_{L^-}(p))\chi(T_{L^-}(p))$. This evaluation is implemented for each type of stratum as follows:

   - 3-dimensional strata: in this case, $N_\epsilon(p)$ is 0-dimensional, thus $\chi(L^-(p)) = \chi(T_{L^-}(p))$ (traditional smooth Morse theory);
   - 2-dimensional strata: in this case, $T_\epsilon(p)$ is the face q itself and $N_\epsilon(p)$ is the line segment $q^\perp$ orthogonal to q going through p. Let nl be the number of ends of $q^\perp$ with lower f values than p, then $\chi(L^-(p)) = 2 - nl$.
   - 1-dimensional strata: these contain no critical point (due to the PT interpolant).
   - 0-dimensional strata: let nl_1, nl_2 and nl_3 be the number of vertices adjacent to p with f values lower than p in each Carte-

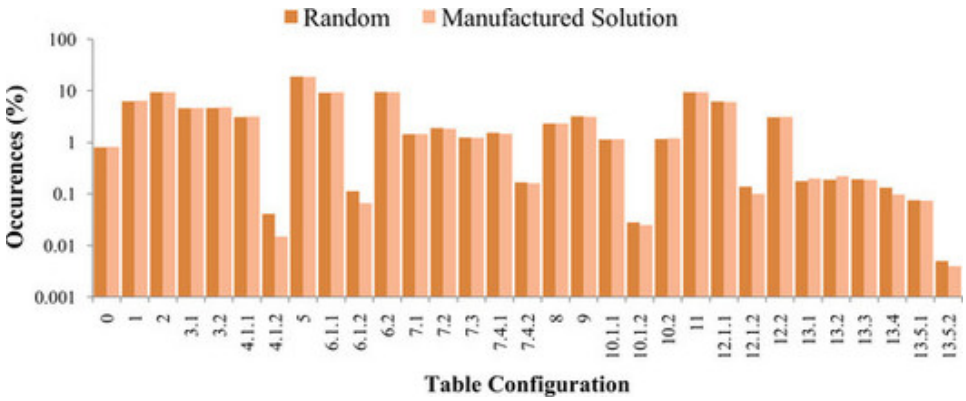


Figure 7.21 – *Rates of occurrence of the Marching Cubes configurations. The horizontal axis shows the cases and subcase numbers for each of the 31 Marching Cubes configurations. The dark bars show the percentage of random fields that fit a particular configuration. The light bars show the percentage of random fields which fit a particular configuration and do not violate the assumptions of our manufactured solution.*

sian coordinate direction. Then: $\chi(L^-(p)) = nl_1 + nl_2 + nl_3 - nl_1(nl_2 + nl_3)$.

$\chi(L^+(p))$ is evaluated symmetrically and $\Delta\chi(p)$ is computed (Equation 7.7).

3. The Euler characteristic χ_α of a manufactured isosurface for isovalue α is then given by:

$$\chi_\alpha = \sum_{p \in C_\alpha} \Delta\chi(p) \tag{7.9}$$

where C_α is the set of critical points sorted by increasing f values such that $f(p) < \alpha, \quad \forall p \in C_\alpha$.

Finally, the Euler characteristic of the surface produced by the isosurfacing code under verification is evaluated (Equation 3). If it equals the evaluation given by the above algorithm, the verification succeeds.

Results

In this section I present experimental results of our framework for the verification of the topological correctness of several isosurfacing programs, implementing various algorithms: VtkMc (VTK implementation of the algorithm described in (MSS94)), Macet (DSS*08), Afront (SSS06), Matlab, SnapMc (RW08), Mc33 (Che95b) (using the implementation provided by Lewiner et al. (LLVT03)), DelIso (DL07) and McFlow (SENS10). Note that only the latter three algorithms (MC33, DelIso and MCFlow) provide theoretical guarantees about the topology of the output.

	Consistency (%)	Betti Numbers (%)			Euler characteristic (%)
		β_0	β_1	β_2	χ
AFRONT	0.0	35.9	22.8	35.9	25.5
MATLAB	19.7	32.2	18.9	20.5	70.3
VTKMC	0.0	27.6	23.2	27.6	70.7
MACET	0.0	54.3	20.9	54.3	100.0
SNAPMC	0.0	45.0	25.4	45.0	72.0
MC33	0.0	2.4	1.1	2.4	5.4
DELISO	19.1	24.4	0.1	20.0	33.2
MCFLOW	0.0	0.0	0.0	0.0	0.0

Table 7.2 – *Rate of mismatches regarding the consistency (manifold test) and the correctness (Betti numbers and Euler characteristic) of the isosurfaces produced by the tested implementations on* 1000 *randomly generated scalar fields.*

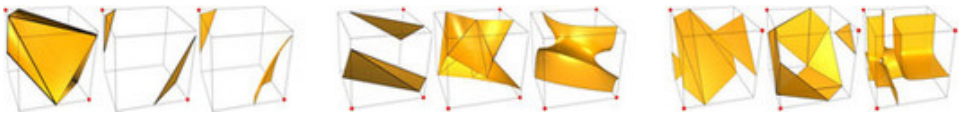


Figure 7.22 – MC33 *mismatch example. From left to right: problem in the case 4.1.2, 6.1.2 and 13.5.2 of the Marching Cubes table (ambiguous cases). Each group of three pictures shows the obtained (left), expected (center) and implicit surfaces (right).*

Our tests consists of one thousand random fields generated in a regular $5 \times 5 \times 5$ grid. Although our methodology may discard certain random fields (presence of ambiguous cells after a large number of refinements, presence of critical points that are too close to edges and faces of the grid, etc), all possible voxel configurations for the trilinear interpolation are still being considered, as illustrated in Figure 7.21.

Table 7.2 shows the results of our verification framework, in terms of rate of mismatch (situations where the topological invariants computed from our verification framework disagree with these of the isosurfacing technique). This table shows that many implementations generate isosurfaces with incorrect number of connected components (β_0), incorrect genus (β_1) and incorrect number of boundary components (β_2). More detailed investigations revealed missing triangles, duplicated vertices, etc. In our initial tests, the MACET implementation failed in all consistency tests. An inspection in the code revealed that boundary cells were not properly traversed. Once that bug was fixed, this implementation started to produce PL manifold surfaces and was successful in the consistency test. In the case of MC33, more detailed investigations on the failure cases revealed a problem with configurations 4, 6 and 13 of the Marching Cubes table (ambiguous cases). Figure 7.22 shows the obtained and expected re-

sults for these configurations. Contacting the author, we found that one of the mismatches was due to a mistake in the implementation of the configuration 13, a non-obvious detail which is not discussed in the original publication (Che95b). In the case of McFlow, our verification framework was applied systematically during its development. Hence all detectable bugs were addressed, resulting in no mismatches.

Concluding remarks

In this section, I presented a verification framework based on simple algorithms for the evaluation of the topological correctness of isosurfacing codes. In particular, our verification procedure easily identified bugs in publicly available isosurfacing codes, including some implementing algorithms providing theoretical guarantees on the topology of the output. Several examples were discussed where these automatically generated diagnoses helped in bug-fixing.

Along with the geometrical accuracy of an isosurfacing code, its topological correctness is of major importance for segmentation tasks in data analysis and visualization. For instance, β_0 directly corresponds to the number of extracted regions of interest and its incorrect evaluation can have drastic implications on the data interpretation in the applications (such as in medical imaging for instance). Thanks to the simplicity of its algorithms, we believe our framework to be easily reproducible and less prone to implementation mistakes than isosurfacing codes themselves. Therefore, thanks to its simplicity and effectiveness, we consider this approach as the reference for the problem of the topological verification of isosurfacing code. Moreover, given the practical importance of topological correctness, we believe our discussion of failure cases and bug-fixes are strong arguments for the usage of our framework during the phase of developments (as done for the McFlow implementation) of any future isosurfacing code.

Perspectives

8

Contents

This chapter presents my vision of the most important challenges that remain to be addressed in Topological Data Analysis for Scientific Visualization. In particular, I describe the ongoing evolution of the constraints and usages in scientific computing, along with their consequences on data analysis. Based on this discussion, I present a list of key problems that I am planning to address in the upcoming years. I also present tentative research directions that I put in the perspective of recently started and upcoming research projects, for which I am the principal investigator at Sorbonne Universites UPMC. The relevance of these research directions will be supported by preliminary results that have been for most of them recently published (GST14, CGT*15, KTCG15) and which show in my opinion great promise for the resolution of the upcoming challenges of Topological Data Analysis.

As illustrated throughout this manuscript, Topological Data Analysis has demonstrated over the last two decades its effectiveness, robustness and practical interest for many data abstraction, exploration and analysis tasks. Moreover, as mentioned at the end of Chapter 3, these techniques gained a sufficient level of maturity such that several of their key algorithms are now available through open-source implementations (Dil07, Tie09, Shi12). This level of maturity is also demonstrated by the numerous collaborations with domain experts for the resolution of problems that go well beyond the sole scope of Computer Science, including molecular chemistry, combustion, cosmology and fluid dynamics for instance. Thus, given this maturity, one can legitimately wonder:

"Is there still any research to be done in this area?"

To answer this question, one needs to observe the ongoing trends in Scientific Computing. Three-dimensional numerical simulation established itself as a necessary tool for knowledge discovery in many fields of science. It enables to evaluate, improve and analyze theoretical models, especially when experimental validation is made difficult for financial, technical or legal reasons. In industry, simulation is ubiquitous in the modeling process of a product.

Traditionally, such simulations are run on High-Performance Computing (HPC) resources while their output (typically a scalar field representing a simulated quantity at a given time-step) is transfered to a remote work station for post-processing purposes: visualization and analysis.

This overall methodology turns out to be incompatible with the characteristics of the upcoming generation of super-computers (expected around 2018) with predicted computing performances at the ExaScale (10^{18} FLoating-point Operations Per Second, FLOPS), since:

1. it will come with unprecedented technical challenges that cannot be addressed by simply extending existing frameworks (Har12) and which will impose new constraints on data analysis algorithms;

2. it will also enable radically novel simulation usages (DA13) which will result in novel types of data to analyze.

As described below, put together, these two challenges require to deeply re-visit the core algorithms of data analysis. This implies a complete *reboot* of the research effort in Topological Data Analysis, which I intend to pursue in the upcoming years.

8.1 Emerging constraints

Current estimations regarding the next generation of super-computers (Har12) expect an increasing imbalance between predicted data throughputs (10^{12} bytes/s (DA13)) and persistent storage (10^{10} bytes/s) or global network bandwidths (most HPC users are located off-site). In other words, data will be produced at significantly higher rates than it can be stored to disk or transferred through the network. In this scenario, traditional off-line post-processing is no longer an option given this increasing bottleneck.

Therefore, to simply avoid this bottleneck, it becomes necessary to move data analysis algorithms as close as possible to their input source (the numerical simulation algorithms). In particular, to minimize the usage of persistent storage, it is necessary that analysis algorithms run on the same hardware as the simulation and that they run during and in synergy with the simulation process. This strategy, often called *In-situ data processing*, imposes three major constraints, described in the following.

8.1.1 Hardware constraints

Supercomputers are built around hardware architectures that differ from commodity computers in several ways. To fully exploit these resources, Topological Data Analysis algorithms need to be adapted to fit the hardware specifications. In particular, a key challenge is to extend them to parallel computing models, including:

1. Shared-memory parallelism with uniform memory access (multi-core environments typically found in any recent computing devices, from handled devices to workstations and supercomputers);

2. Shared-memory parallelism with non-uniform memory access (multi-processor environment typically found in high-end workstations and supercomputers);

3. Distributed memory parallelism (multi-node environments typically found in supercomputers).

Each of these types of parallelism comes with increasing difficulty regarding memory transfer management.

Adapting Topological Data Analysis algorithms to these three types of parallelisms is a major algorithmic challenge since these approaches are intrinsically sequential, use only few floating point arithmetic operations

(they are mostly based on graph traversal sub-routines) and always rely at some point on a global access to the data (in particular to enable their global consistency).

Moreover, another major concern to be addressed for the upcoming generation of supercomputers is energy consumption (Har12). In particular, one well-accepted way to reduce the energy consumption related to the execution of a program is to reduce its memory usage and transfer. In terms of algorithms, this is also a major algorithmic challenge for Topological Data Analysis, as most of its algorithms are memory bound.

Therefore, to enable Topological Data Analysis for the upcoming generation of supercomputers, I intend to deeply revisit most of its algorithms to develop memory-efficient parallel algorithms.

Preliminary results

To initiate this research effort, I first focused on the case of the Contour Tree and the Reeb graph, in collaboration with Pierre Fortin (assistant professor at UPMC and expert in HPC) in the context of Charles Gueunet's Master thesis. While several approaches attempted to parallelize the Contour Tree construction algorithm (PCM03, MDN12, MW13, LPG*14, AN15), these techniques only considered the case of a simpler variant (the join or split tree) or relied on a computation based on monotone paths (which drastically restricts the usage of the output data structures in applications, preventing for instance data segmentation features).

For the problem of Contour Tree parallel computation in a shared-memory context, we introduced a new and simple data-division scheme to balance the input data-set in between the different threads. In particular, this strategy makes the stitching of the intermediate results computed from the different threads into one global structure much easier and more efficient than previous approaches. Our strategy also directly enables the parallelization of persistence-driven simplification, a problem which has not been addressed so far to the best of our knowledge. Our preliminary results demonstrate virtually linear scalability with the number of cores for simple scalar fields and we are currently addressing issues related to non-uniform memory access effects on multi-processor architectures for more complex scalar fields.

In the future, we plan to extend this kind of data-division approach to account for non-simply connected domains, hence generalizing our technique from Contour Trees to Reeb Graphs. Further, I will continue this

research efforts by revisiting other Topological Analysis Algorithms in the perspective of shared-memory and distributed parallelism.

8.1.2 Software constraints

In-situ data post-processing could be achieved either in *(i)* synchronous or *(ii)* asynchronous manner. In the first case *(i)*, the data production process (i.e. the simulation code) is temporarily put on pause to make the computing resources available to the in-situ post-processing program. This approach requires intrusive scheduling policies to be implemented in the simulation code (to observe periodic pauses) and also imposes to store in memory all of the data that will be required by the post-processing program. A more flexible and less memory-demanding strategy is the asynchronous data post-processing *(ii)*, where data is post-processed *on-the-fly* as it is produced. Unlike the synchronous mode, this strategy does not require to store in memory large portions of the generated data and is transparent in terms of scheduling from the simulation code's perspective. While this strategy facilitates the deployment of in-situ post-processing into existing simulation codes, it shifts the scheduling constraints to the post-processing program.

In particular, in the asynchronous framework, to optimize the scheduling between data generation and post-processing, one should be able to impose resource budgets (both in terms of memory and computation time) on the data-analysis algorithms. For instance, these should be able to process data as long as it is maintained in memory and produce a usable output when it is deleted.

These software management constraints constitute a major algorithmic shift for Topological Data Analysis, since it requires to develop *best-effort* algorithms whereas only exact algorithms have been designed by the community. In other words, in this setting, one would like to develop Topological Data Analysis algorithms capable of approximating their output and refining it progressively as long as the computation time budget has not expired. Note that this *coarse-to-fine* strategy is the complete opposite of current algorithms (which are *fine-to-coarse*), where the exact solution is first computed and then progressively simplified with persistent homology mechanisms (see Chapter 3). Designing *best-effort* Topological Data Analysis algorithms is one of the key problems I will address in the upcoming years.

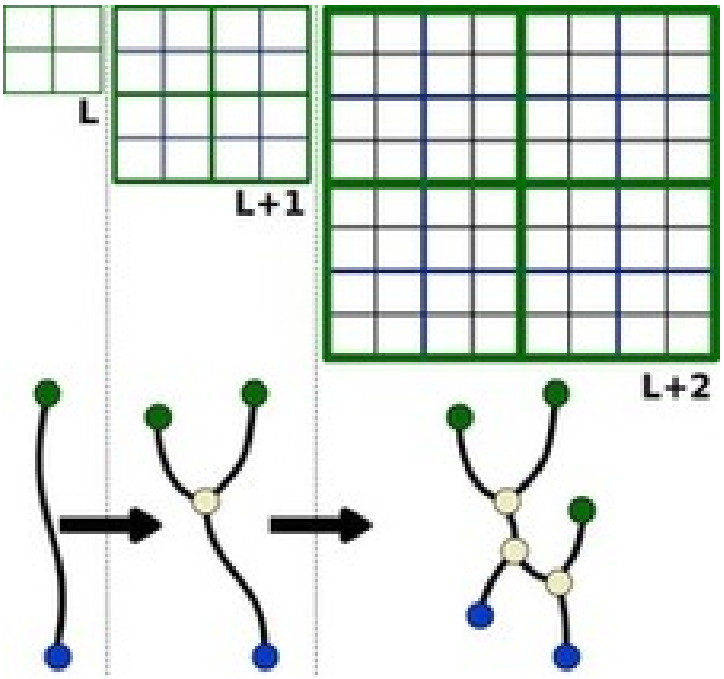


Figure 8.1 – *Progressive Contour Tree Computation. A first computation is performed at the coarsest level (L) with existing algorithms, while new algorithms need to be designed to efficiently perform only the required updates (dark arrows) to obtain the Contour Tree of the next hierarchy levels ($L+1$, $L+2$).*

Research directions

A promising research direction to develop best-effort Topological Data Analysis algorithms is to consider hierarchical data representations, as illustrated in Figure 8.1 for the Contour Tree in the case of a hierarchy of 2D regular grids. Then, while existing algorithms can be used for the coarsest level of the data hierarchy, new algorithms need to be designed to efficiently update the output data-structure at the next hierarchy levels by only performing the required updates.

Note that, if one considers a min-max data hierarchy (storing for each vertex of a resolution the minimum and maximum values of the vertices it represents at the next resolution), this progressive data computation can be viewed as an uncertain process, where the current resolution describes an error-affected representation of the data whose error is reduced along the data hierarchy.

As further described in the next section, I recently introduced an approach for the generalization of the Join Tree (a simpler variant of the Contour Tree) for uncertain scalar fields (GST14). Figure 8.2 illustrates the computation of this structure for 4 different hierarchy levels (from left to right: coarsest to finest). In particular, blue and green regions denote re-

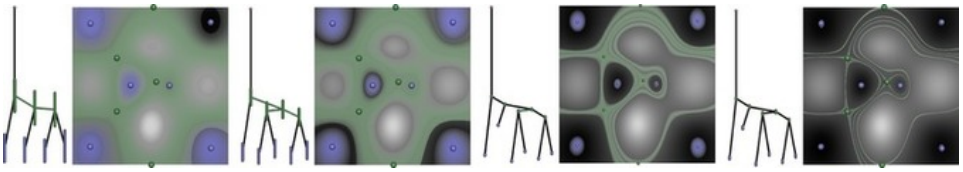


Figure 8.2 – *Progressive Join Tree construction illustrated as an uncertain process. The minima and saddles of the finest resolution of the data (gradient of gray in the background) are shown with blue and green spheres respectively. For each data-resolution (from left to right: coarsest to finest) our uncertain topological analysis has been run. The predicted regions for the appearance of minima and saddles are shown in blue and green respectively.*

gions of possible appearance of minima and saddles at the finest hierarchy level. As showcased in this illustration, the Join Tree not only gets more accurate along the hierarchy from a topological point of view (number of critical points), but also from a geometrical point of view with more and more refined predictions of the locations of the critical points at the finest level of the hierarchy. Therefore, to accompany *best-effort* algorithms with prediction on the accuracy of the output, I intend to extend my uncertain data processing algorithm (GST14) with update mechanisms to efficiently progress from a hierarchy level to another.

This research direction is the core topic of Charles Gueunet's Ph.D. thesis, that will start in early 2016 under my supervision in the framework of a CIFRE industrial partnership with Kitware S.A.S. (the French subsidiary of Kitware Inc.). During this thesis, we will address the problem of progressive topological abstraction computation with accuracy predictions, starting from the Join Tree and then generalizing to more complex topological abstractions.

8.1.3 Exploration constraints

My discussion about In-Situ data processing focused so far on the integration of Topological Data Analysis algorithms in an HPC context, mostly for data abstraction and analysis purposes, assuming these algorithms to run in batch-mode, alongside the simulation. Therefore, this discussion left the question of interactive exploration in an HPC context open.

An accepted strategy to offer interactive exploration capabilities in an HPC context is in-situ data reduction for a posteriori interactive exploration on a remote computer (or *post-mortem exploration*). Then, the challenge consists in efficiently reducing the generated data such that:

- the reduced data has a size that becomes manageable for persistent

storage or network transfer (hence enabling a posteriori exploration on a remote computer);

- the reduced data still exhibits precisely the features of interest in the data.

A first obvious direction to address in-situ data reduction for post-mortem exploration is data lossy compression.

Another, orthogonal, approach for data reduction consists in anticipating the possible user interactions and pre-computing in-situ the result of each interaction. In particular, given a list of parameters that a user can control in an interactive task, the idea here is to sample this parameter space and pre-compute in-situ the visual outcome of each parameter combination. For instance, this idea is the core strategy of ParaView Cinema (Kit14). Given a visualization pipeline for a data-set and its list of parameters (camera position, view angle, color map, etc.), this system samples this parameter space and for each parameter combination generates a corresponding 2D rendering. Then the output collection of 2D renderings (which is orders of magnitude smaller than the actual data) can be interactively explored on a remote computer, with interactions that emulate changes in camera position, view angle, color map, etc.

While the example of ParaView cinema addresses the actual post-mortem visualization of the data, it does not address post-mortem interactive feature extraction and exploration. Thus, in the upcoming years, I would like to extend this strategy to Topological Data Analysis algorithms for interactive feature extraction and exploration tasks. This research problem requires to re-visit each interactive technique based on topological abstractions and derive the appropriate output data encoding given the parameters of the interaction.

Preliminary results

As described in Chapter 6, along with LLNL collaborators I already started to explore this research direction for the interactive post-mortem exploration of features in the context of combustion simulations (BWT*09, BWT*11). In this work, we showed that a concise encoding of the possible segmentations of the data by the split tree enabled a post-mortem interactive exploration and tracking of the flames present in the simulation. In the future, I would like to extend this strategy in multiple ways:

- Extending this approach and the segmentation encoding to only store on disk the finest segmentation and enable post-mortem progressive simplification;
- Enriching this strategy with optional lossy reduction by pre-simplifying the segmentations in-situ up to a user tolerance;
- Enriching this approach with out-of-core capabilities on the client side, to enable post-mortem exploration even with low-memory client workstations;
- Extending this approach to other, more complex, topological abstractions (Contour Tree, Reeb Graph, Morse-Smale complex);
- Specializing this general strategy to specific application scenarios;
- Extending this approach to unstructured meshes.

These research directions constitute the core topic of Maxime Soler's Ph.D. thesis, that should start in 2016 under my supervision in the framework of a CIFRE industrial partnership with Total.

8.2 Emerging data types

In addition to the emergence of new algorithmic constraints (previous section), the current increase of the performance of supercomputers allows new simulation usages, yielding the emergence of new data types.

As described below, these new data types cannot be handled by current analysis algorithms, which also requires to revisit Topological Data Analysis algorithms to take them into account.

8.2.1 Multivariate data

Given the recent HPC performances, it becomes now possible to model complex macroscopic processes by jointly simulating multiple physical phenomena (for instance: joint thermodynamic, computational fluid dynamics and material resistance simulations). In this context, a given simulation generates, for a given PL manifold $\mathcal{M}$, a *family* of n scalar functions that represent drastically (and physically) different quantities (temperature, flow vorticity, material stress, etc.):

$$f : \mathcal{M} \rightarrow \mathbb{R}^n \tag{8.1}$$

This variability leads concretely to scalar fields having drastically different range values and dynamics. The joint analysis of several scalar fields defined on a common domain is therefore a major challenge that needs to be addressed to identify and quantify possible geometrical correlations between quantities.

However, traditionally, Topological Data Analysis only deals with the analysis of one scalar function defined on a single geometry. Therefore, in this topic, I will extend the concepts of Topological Data Analysis to *multivariate* scalar functions. This effort will be accompanied with the design of algorithms that are efficient in practice for the construction and simplification of these generalized topological abstractions, and their exploitation in specific application problems.

Preliminary results

In the context of a collaboration with the University of Leeds, we are currently working on the design of an algorithm with efficient practical performances for the problem of Reeb space computation (EHP08). This construction generalizes the notion of Reeb graph to multivariate scalar functions by tracking connected components of *fibers* (multivariate analogs of level-sets). While an algorithm has been described for its approximation (CD14), no algorithm with efficient practical behavior has been documented for its exact computation and simplification.

We addressed this problem with a new output-sensitive algorithm for the exact construction of Reeb spaces of piecewise linear bivariate scalar fields on tetrahedral meshes:

$$f : \mathcal{M} \rightarrow \mathbb{R}^2 \tag{8.2}$$

In particular, thanks to its output-sensitive nature, this algorithm is fast in practice and it can be shown that its computation time requirement is equivalent to that of the theoretical algorithm by Edelsbrunner et al. (EHP08) only in the worst-scenario (random data) and is faster otherwise. I am currently finishing the implementation of this algorithm along with simplification capabilities. As Reeb graphs and Contour Trees offered new analysis and interaction capabilities for scalar fields (see Chapter 3), I am convinced that Reeb spaces (and in particular our efficient algorithm) will enable a wide range of analysis methods for bivariate data, hence initiating a new line of research. Therefore, in the upcoming years, I will work on the extensions of the applications of the Reeb graph in Scien-

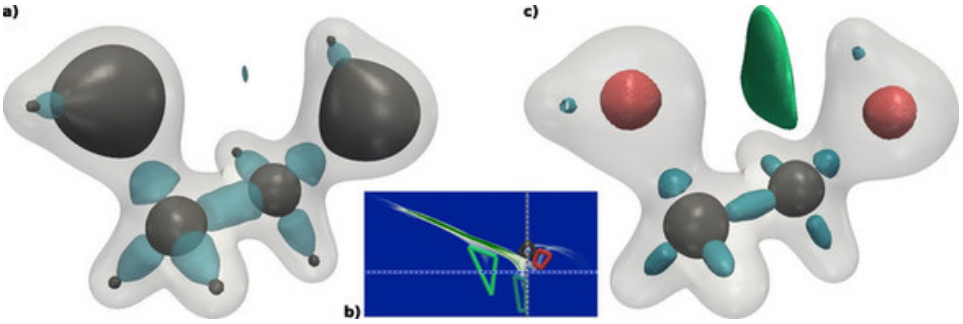


Figure 8.3 – *Isosurfaces (a) and Fiber surfaces (c) of a bivariate field representing chemical interactions within an ethane-diol molecule ((b): continuous scatter plot, X: electron density, Y: reduced gradient). While isosurfaces of the electron density capture regions of influence of atoms ((a): grey), they do not distinguish atom types. Similarly, isosurfaces of the reduced gradient capture regions of chemical interactions ((a): blue) but do not distinguish covalent from non-covalent interactions. In contrast, polygons isolating the main features of the continuous scatter plot (b) yield fiber surfaces (c) distinguishing atom types (red and grey) as well as interaction types (blue and green).*

tific Visualization to the case of bivariate data. In the following, I further motivate the applicative interest of such analysis capabilities.

In the process of defining our fast algorithm for Reeb space computation, we needed to introduce a novel construction called *Fiber Surfaces*, which are pre-images of PL 1-manifolds through bivariate functions. Surprisingly we discovered that this novel construction was somehow implicitly known by the volume rendering community for multi-dimensional transfer function definition. However, this community could only visualize a volume rendering of these fiber surfaces and no algorithm was documented to extract them geometrically. We therefore presented a simple approach to compute fiber surfaces and showed the applicative interests of such constructions for data segmentation purposes in various application fields (CGT*15). For instance, Figures 8.3 and 8.4 demonstrate the superiority of fiber surfaces over isosurfaces for the user-driven segmentation of simulated or acquired data.

However this first algorithm was slow in practice for large data-sets and was only approximate, as illustrated in Figure 8.5. Therefore, we introduced a second algorithm for the exact computation of fiber surfaces as well as several acceleration mechanisms (generalized from isosurface extraction acceleration) which enabled an interactive exploration of the space of fiber surfaces (KTCG15). We consider this latter algorithm as the reference for the problem of efficient and exact fiber surface computation in bivariate scalar fields and we released a VTK-based open source implementation of it in the hope of a rapid uptake of this method.

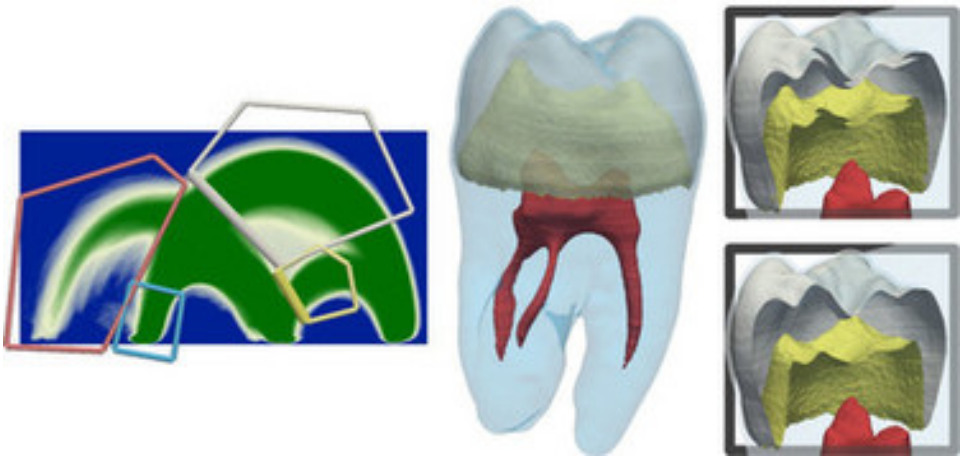


Figure 8.4 – *Fiber surface extraction on an acquired data set (CT-scan). The considered bivariate data is given by the acquired value (X axis of the continuous scatter plot, left) and the magnitude of its gradient (Y axis, left). By manually contouring the main features of the continuous scatter plot (left), the user can easily extract with the corresponding fiber surfaces the boundary of the regions of interest of the volume (middle: pulp in red, dentin in blue, enamel in white, boundary between the dentin and the enamel in yellow). Thanks to our new algorithm, fiber surfaces can also be computed for non-closed polygons (thicker edges on the left, fiber surfaces in the bottom zoom-in view on the right).*

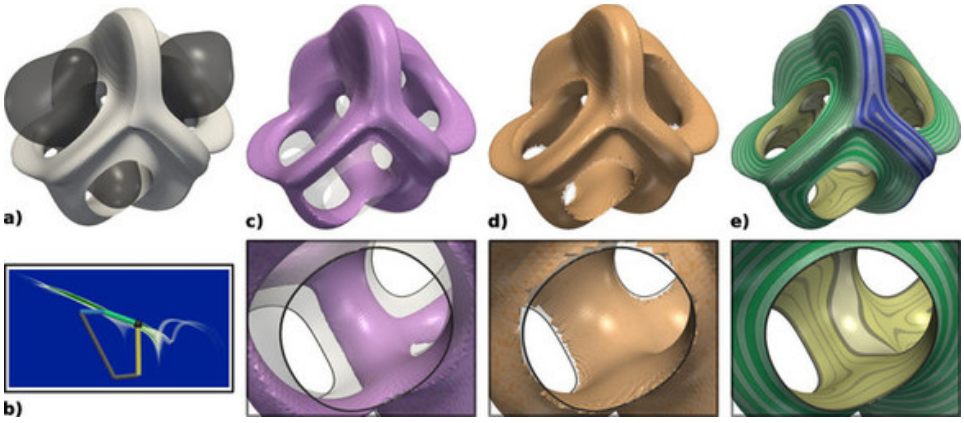


Figure 8.5 – *Fiber surface extraction on a bivariate field (electron density and reduced gradient) representing chemical interactions within an ethane-diol molecule (dark surface in (a)). Fiber surfaces are defined as pre-images of polygons drawn in range space (i.e. the continuous scatter plot (b)). The first algorithm for their computation (CGT*15) relies on a distance field computation on a rasterization of the range. While increasing the raster resolution results in more accurate fiber surfaces ((c): 162 , (d): 10242), even for large resolutions, the distance field intrinsically fails at capturing sharp features of the fiber surface (here polygon bends in the range, black sphere (b)), as showcased in the zoom-views (bottom) where the corresponding fibers are displayed with black curves. Our new approach (KTCG15) introduces the first algorithm for the exact computation of fiber surfaces on tetrahedral meshes. It accurately captures sharp features (e) and enhances fiber surfaces with polygon-edge segmentation (colors in (b) and (e)) and individual fibers (e, bottom) to better convey the relation between fiber surfaces and range features.*

As illustrated in Figures 8.3, 8.4 and 8.5, fiber surface extraction currently requires manual intervention based on the perceived features in the continuous scatter plot (BW08) of the data. However, as illustrated in the case of the two oxygen atoms in Figure 8.3 (two red surfaces, right), several distinct features may exist in 3D for the same location in the continuous scatter plot (single red curve, middle). To disambiguate these configurations and to further help the user explore the continuous scatter plot, I plan to investigate in the future the usage of my Reeb space computation algorithm for the automatic feature segmentation and simplification in bivariate data. Indeed, the Reeb space is guaranteed by definition to segment the 3D space into regions where fibers (and fiber surfaces) are made of a single connected component. Such a contribution would have an impact on any visualization task dealing with bivariate data, such as feature extraction or transfer function design. However, to make such a topological abstraction useful in practice, as it was the case for scalar data, persistent homology concepts need to be generalized to bivariate data to allow for efficient simplification algorithms. Thus, in the upcoming years, based on my ongoing results on Reeb spaces, I plan to generalize Topological Data Analysis to bivariate data by extending persistent homology concepts as well as applications of the Reeb graph in visualization to the bivariate case. In particular, I intend to study the application of these algorithms in collaboration with domain experts at Sorbonne Universites UPMC in scientific fields going from computational chemistry to astrophysics.

8.2.2 Uncertain data

A physical model is often dictated by a number of parameters (initial conditions, boundary conditions, etc.). Given the recent HPC advances, the fine sampling of this parameter space becomes feasible (yielding one simulation output per combination of parameters). This process, called parameter study, is central to the understanding of the uncertainty that accompanies any physical process. For instance, it enables to identify parameter ranges for an efficient and safe functioning of a complex system. This type of simulation also generates, given a common domain, a *family* of n scalar fields, that model an uncertain process:

$$f : \mathcal{M} \rightarrow \mathbb{R}^n \tag{8.3}$$

This collection of n scalar fields is often called an *ensemble data-set* and each scalar field a *member* of this ensemble. Alternatively, each scalar field

can be seen as an *observation* of an *uncertain scalar field*, which maps each point of the domain to a random variable. In this latter context, the data is typically represented for each vertex of the domain by a probability density function (estimated for instance by a histogram). Analyzing this family of scalar fields (ensemble members or observations) as a whole to identify, extract and understand the conditions of appearance of features is a major upcoming challenge in visualization and analysis.

For uncertain data, the number n of considered scalar fields is typically much higher than in the case of multivariate data (previous section). Moreover, from a theoretical point of view, the topological analysis of multivariate data seems to have an applicative interest only when the dimension of the range is lower than the dimension of the domain (typically three). For instance, as of $n = 3$, the Reeb space of a generic multivariate scalar field defined on a PL 3-manifold $\mathcal{M}$ is $\mathcal{M}$ itself. Thus, for n values beyond the dimension of the domain, another direction needs to be considered. Therefore, the topological analysis of multivariate fields seems of little importance for the processing of uncertain data. Instead, in this topic, I will address the problem of generalizing the constructions of Topological Data Analysis to uncertain scalar fields (that map each point of the domain to a random variable).

Preliminary results

In the context of an exploratory project for which I was the principal investigator (called "UnTopoVis", national funding, Digiteo), I introduced the first non-local, combinatorial characterization of critical points and their global relation in 2D uncertain scalar fields (GST14). The characterization is based on the analysis of the support of the probability density functions of the input random variables. Given two scalar fields representing reliable estimations of the bounds of this support (noted $f^- : \mathcal{M} \to \mathbb{R}$ and $f^+ : \mathcal{M} \to \mathbb{R}$), based on the observation that their sub-level sets were point-wise nested, we described sufficient conditions (Figure 8.6) for the appearance of *mandatory minima*. This latter construction generalizes the notion of local minimum from PL scalar fields to uncertain scalar fields:

Definition 65 (Mandatory minimum) *A* mandatory minimum *M is a minimal connected component $C \subset \mathcal{M}$ with a minimal interval $I \subset \mathbb{R}$ such that any realization field g (a scalar field that maps each vertex to a realization of its random variable) admits at least one minimum $m \in C$ with $g(m) \in I$. C is called the* critical component *of M and I its critical interval.*

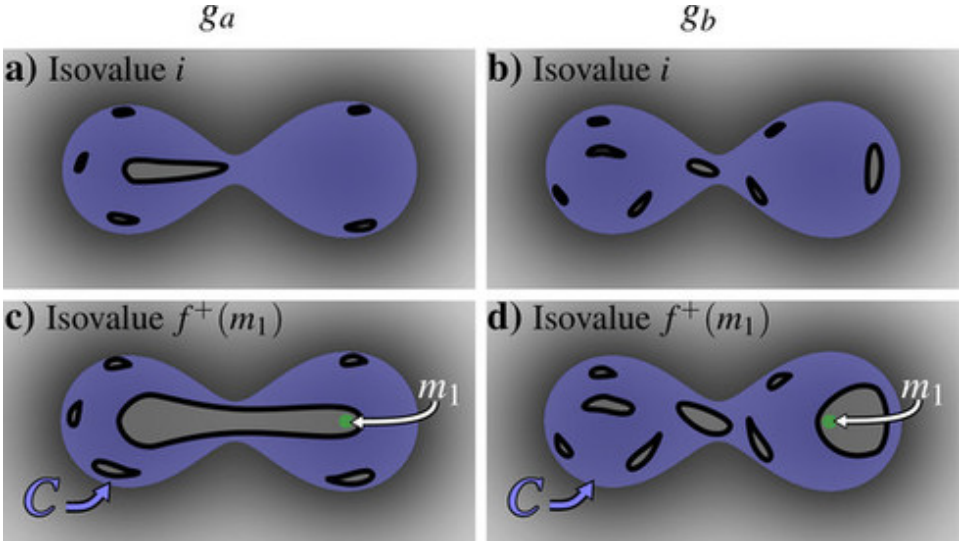


Figure 8.6 – *Sub-levels of the lower (f^-, blue) and upper (f^+, green) bounds of an uncertain 2D scalar field for increasing isovalues (top to bottom). The sub-level sets of two realization fields of the uncertain data are shown in grey (left: g_a, right: g_b). All points in the blue region have a non-zero probability to get a value lower than the current isovalue, while all the points in the green region have a probability of* 1 *to be lower. Therefore, the appearance of a unique local minimum m of f^+ (in green) within a connected component C of the sub-level set of f^- is a sufficient condition for the appearance of a* mandatory minimum*, whose critical component is C. Such a configuration indeed implies the existence of at least one connected component of sub-level set of any realization field g, included in C and including m (bottom row). Such a component implies the existence of at least one local minimum of g in C.*

Note that this construction nicely generalizes the notions of critical points and critical values to critical components and critical intervals. The notions of *mandatory maxima*, *mandatory join saddles* and *mandatory split saddles* are defined similarly.

Thanks to the specification of the appearance conditions of mandatory critical points, we described a combinatorial algorithm for their extraction. This strategy hence identifies spatial regions and function ranges where critical points have to occur in any realization of the input uncertain data, as illustrated in Figure 8.7. In other words, these regions form the common topological denominator to all realizations (or observations) of an uncertain scalar field: they describe the common topological features that appear in all of these scalar fields. From an application point of view, this approach enables to predict the location and the minimum number of vortices for instance in uncertain flow simulations, as illustrated in Figure 8.7.

Our algorithm provides a global pairing scheme for mandatory critical points which is used to construct mandatory join and split trees. These

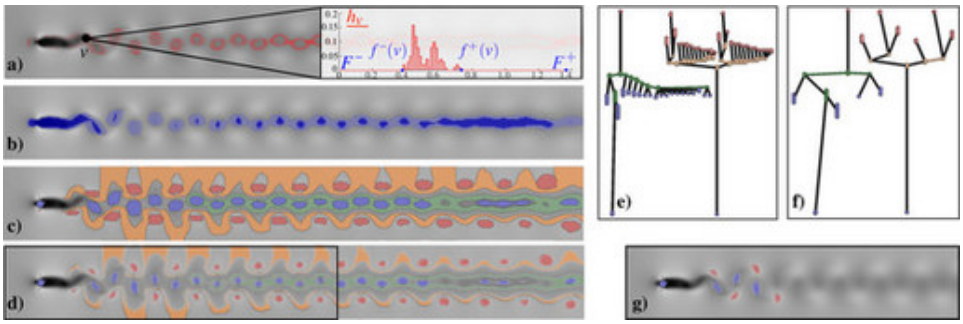


Figure 8.7 – *Mandatory critical points of the velocity magnitude of the uncertain Kármán vortex street. a) Each vertex v is assigned with a histogram h_v estimating its probability density function. The shades of red show the point-wise probability for the isovalue 0.85. b) The support of h_v is visualized by the lower (f^-, light blue) and upper (f^+, dark blue) bound fields. c) depicts the mandatory critical points (blue: minimum, green: join saddle, yellow: split saddle, red: maximum), d) illustrates the spatial uncertainty within the components. e) shows the mandatory join/split tree, and f) and g) the simplified visualization.*

trees enable a visual exploration of the common topological structure of all possible realizations of the uncertain data. To allow multi-scale visualization, we introduce a simplification scheme for mandatory critical point pairs revealing the most dominant features. Our technique is purely combinatorial and handles parametric distribution models and ensemble data. It does not depend on any computational parameter and does not suffer from numerical inaccuracy or global inconsistency. The algorithm exploits ideas of the established join/split tree computation. It is therefore simple to implement, and its time complexity is output-sensitive. Experiments demonstrated the accuracy and usefulness of our method on synthetic and real-world uncertain data-sets. Thanks to its robustness and multi-scale nature, we consider this approach as the reference algorithm for the problem of mandatory critical point extraction in 2D uncertain scalar fields.

Despite this strong result, this first attempt at extending Topological Data Analysis to uncertain data raised even more questions than answers. In particular, despite their strong applicative interest, the topological features that are common to all realizations of an uncertain process (i.e. *that have a probability of appearance of 1*) only constitute a sub-set of the features users are interested in. From a theoretical point of view, a natural question that arise is:

"What about the critical points with a probability of appearance lower than 1?"

This question has strong applicative implications. Often, the phenom-

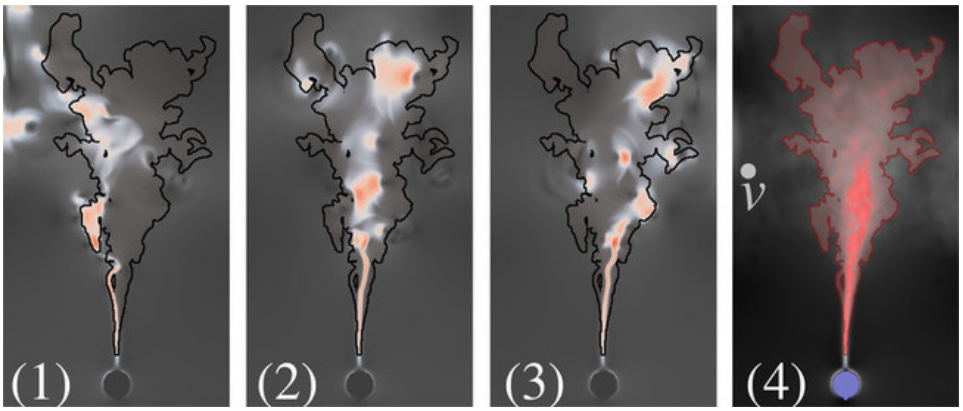


Figure 8.8 – *Uncertain scalar field describing the velocity field caused by a heat source (bottom). Our algorithm for mandatory critical point computation extracts a dominant maximum (in red, rightmost) which describes the cone through which the flow travels (and attain velocity maxima) for all observations. However, as illustrated on the left with* 3 *observation fields (*1 *to* 3*), the flow describes three distinct regimes (leftward, center and rightward trajectories) which are not identified and characterized by our approach.*

ena under investigation can reveal distinct regimes and it is important to understand the probability of appearance of these regimes as well as the conditions (sets of parameters) for their appearance. Figure 8.8 shows such an example where the uncertain data clearly exhibits three distinct regimes that are not identified and characterized by our approach.

In the upcoming years, I therefore intend to address the problem of ensemble data (or uncertain data) topological classification for regime extraction and characterization. As described above, such a research direction is important because of its applicative context (parameter studies) where such analysis capabilities are necessary to abstract, interact with and analyze uncertain data. From a technical perspective, this problem is highly challenging for several reasons. First, it requires to bridge the gap between algorithmic techniques coming from different communities (Topological Data Analysis and Machine Learning). Second, given the size of the ensemble data-sets to consider, it is likely that such analyses can only be performed in-situ. Last, it requires to address sub-problems which have not (or only partially) been addressed by the community. In particular, to develop relevant models for the topological classification of ensemble data-sets, I plan to address the following sub-problems:

1. Designing stable and discriminative distance metrics between Topological Abstractions as well as efficient algorithms for their computation;

2. Designing efficient and relevant algorithms for the clustering of Topological Abstractions given the above distance metrics;

3. Analyzing the parameter space of parameter studies in the light of the topological clustering of their observations and specializing this overall strategy to several application scenarios.

I am currently starting this research effort, in particular by investigating optimal transportation techniques applied to Topological Data Analysis, in the framework of the AVIDO research project, for which I am the local investigator at Sorbonne Universites UPMC and for which I will hire an engineer and a post-doctoral researcher. This project brings together academic researchers (UPMC and INRIA) as well as domain experts in the framework of an industrial partnership with EDF (the major French electricity provider) and Kitware S.A.S. to address the problem of the in-situ analysis of parameter studies. In particular, thanks to its concrete use-case scenarios, this project will provide a unique environment for the development and experimentation of my algorithms.

Conclusion

9

In this thesis, I summarized my key research results since my Ph.D. defense in 2008. After a concise tutorial and survey on Topological Data Analysis for Scientific Visualization (Chapter 3), I presented my contributions to this field, in particular in each of the following topics:

1. **Abstraction:** In this topic, I presented my algorithmic contributions to the computation of Topological Abstractions of raw scalar data. In particular, I described a general algorithm for the topological simplification of scalar data on surfaces (TP12) and showed that, when used as a pre-processing step, it could drastically improve the time performance of topological abstraction computation algorithms. Thanks to its generality, robustness, ease of implementation and practical performances, we consider this algorithm as the reference for the problem of topological simplification of scalar data on surfaces. Next, I presented an efficient algorithm for the computation of the Reeb graph of scalar data defined on PL 3-manifolds in $\mathbb{R}^3$ (TGSP09). This approach described the first practical algorithm for volumetric meshes, with virtually linear scalability in practice and up to 3 orders of magnitude speedups with regard to previous work. Such an algorithm enabled for the first time the generalization of contour-tree based techniques to general, non simply-connected volumes. We considered this algorithm as the reference for the problem of Reeb graph computation on volumes until an optimal time complexity algorithm was introduced three years later (Par12).

2. **Interaction:** In this topic, I presented my algorithmic contributions for the interactive editing of Topological Abstractions for data exploration and user-driven segmentation tasks. In particular, I described interactive algorithms based on the Reeb graph for the topological simplification of surfaces, both on-line (TGSP09) and in an in-situ

context (STK*09). Moreover, I presented two algorithms for the interactive editing of Topological Abstractions (the Morse-Smale complex (GGL*14) and the Reeb graph (TDN*12)) to enable user-driven topological data segmentation. In particular, these algorithms enable to incorporate user knowledge in segmentation tasks where features of interest are aligned with the gradient of the data (Morse-Smale complex) or with its level sets (Reeb graph).

3. **Analysis:** In this topic, I presented my contributions in the specialization of Topological Data Analysis techniques to specific application problems in combustion (BWT*09, BWT*11) and chemistry (GABCG*14). In particular I showed how standard Topological Data Analysis could be adapted to extract and analyze features of interest in these fields. These two applications demonstrated that thanks to their robustness and relevance, Topological Data Analysis techniques could address diversified scientific issues well beyond the sole scope of Computer Science.

4. **Related problems:** In this topic, I described related problems that have been addressed with a solution derived or inspired from Topological Data Analysis for geometry processing tasks (such as scalar field design with topological guarantees (TP12) or interactive quadrangulation design (TDN*12, TDN*11)), interactive panorama stitching (STP12, PST*13, PST*15) or isosurface topological verification (ENS*12). For two of these problems, we consider the developed algorithms to be reference solutions (STP12, ENS*12).

All of these results were obtained in collaboration with several research groups (University of Utah, Lawrence Livermore National Laboratory, Lawrence Berkeley National Laboratory, Universidade de Sao Paulo, New York University, Sorbonne Universites, Clemson University, University of Leeds) as well as students whom I informally or formally advised.

In the process of introducing the key concepts, algorithms and applications of Topological Data Analysis (through the tutorial Chapter 3 and the description of my contributions, Chapters 4, 5, 6, and 7), I emphasized the robustness, time-performance and efficiency of such algorithms for several data abstraction, exploration and analysis tasks. I also underlined the maturity attained by these techniques, as demonstrated by the emergence of open-source implementations of these algorithms, to which I partially

contributed. This level of maturity is also demonstrated by the numerous collaborations with domain experts for the resolution of problems that go well beyond the sole scope of Computer Science, including combustion, molecular chemistry, cosmology, and fluid dynamics for instance.

Despite this level of maturity, I described in Chapter 8 the challenges that I believe to be the most important ones for Topological Data Analysis in the future. In particular, I discussed how these challenges were greatly related to the upcoming generation of supercomputers, which will *(i)* impose new constraints on analysis algorithms and which will *(ii)* enable the generation of new data types to analyze. Put together, these two challenges require to deeply revisit Topological Data Analysis algorithms and to reboot the research effort made by the community in this area over the last two decades.

In particular, due to the input-output bandwidth bottleneck, in-situ data processing will become inevitable. This requires Topological Data Analysis techniques to adapt to new hardware constraints (massively parallel and distributed architectures), new software constraints (with best effort computations, in order to ease the scheduling between data generation and post-processing) and exploration constraints (requiring to define in-situ data reduction algorithms for post-mortem interactive exploration). For each of these constraints, I described the consequent algorithmic challenges that I intend to address in the upcoming years.

Furthermore, the computing performances of the next generation of supercomputers will allow for a systematic usage of parameter studies or multi-physics simulations. Both of these usages will make emerging data types much more prominent such as *(i)* multivariate data and *(ii)* uncertain data. Generalizing Topological Data Analysis to these two data types requires not only a major algorithmic effort but also an important theoretical investigation about the extension of Morse theory concepts. I intend to address these challenges in the upcoming years, in particular in the context of recently started research projects (including industrial partnerships) for which I am the principal investigator. These projects will allow me to recruit and train younger researchers to join and participate to this research effort. As described through Chapter 8, my preliminary results in this topic (CGT*15, KTCG15, GST14) raised even more questions than answers, but still enabled to identify promising research directions to address these challenges in the future.

Index

Bibliography

[ABD*13] Attali D., Bauer U., Devillers O., Glisse M., Lieutier A.: Homological reconstruction and simplification in R3. In *Proc. of ACM Symposium on Computational Geometry* (2013). (Cited page 88.)

[AEHW04] Agarwal P. K., Edelsbrunner H., Harer J., Wang Y.: Extreme elevation on a 2-manifold. In *Proc. of ACM Symposium on Computational Geometry* (2004). (Cited pages 91 and 95.)

[AGH*09] Attali D., Glisse M., Hornus S., Lazarus F., Morozov D.: Persistence-sensitive simplification of functions on surfaces in linear time. In *TopoInVis Workshop* (2009). (Cited pages 61 and 85.)

[AN15] Acharya A., Natarajan V.: A parallel and memory efficient algorithm for constructing the contour tree. In *Proc. of PacificVis* (2015). (Cited page 197.)

[Bad94] Bader R.: *Atoms in Molecules: A Quantum Theory*. Oxford University Press, 1994. (Cited pages 148 and 159.)

[Ban70] Banchoff T. F.: Critical points and curvature for embedded polyhedral surfaces. *The American Mathematical Monthly* (1970). (Cited pages 49 and 60.)

[BC95] Bedat B., Cheng R. K.: Experimental study of premixed flames in intense isotropic turbulence. *Combust. Flame 100* (1995), 485–494. (Cited page 133.)

[BEHP03] Bremer P.-T., Edelsbrunner H., Hamann B., Pascucci V.: A multi-resolution data structure for 2-dimensional Morse functions. In *IEEE Visualization* (2003), pp. 139–146. (Cited page 65.)

[BEHP04] Bremer P.-T., Edelsbrunner H., Hamann B., Pascucci V.: A topological hierarchy for functions on triangulated sur-

faces. *IEEE Transactions on Visualization and Computer Graphics 10* (2004), 385–396. (Cited page 86.)

[BGSF08] Biasotti S., Giorgio D., Spagnuolo M., Falcidieno B.: Reeb graphs for shape analysis and applications. *Theoretical Computer Science* (2008). (Cited page 64.)

[BK04] Boykov Y., Kolmogorov V.: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2004). (Cited page 175.)

[BLW12] Bauer U., Lange C., Wardetzky M.: Optimal topological simplification of discrete functions on surfaces. *Discrete and Computational Geometry* (2012), 347–377. (Cited pages 61 and 85.)

[BR63] Boyell R. L., Ruston H.: Hybrid techniques for real-time radar simulation. In *Proc. of the IEEE Fall Joint Computer Conference* (1963). (Cited page 62.)

[BW08] Bachthaler S., Weiskopf D.: Continuous scatterplots. *IEEE Transactions on Visualization and Computer Graphics (Proc. of IEEE VIS)* (2008). (Cited page 206.)

[BWT*09] Bremer P., Weber G., Tierny J., Pascucci V., Day M., Bell J.: A topological framework for the interactive exploration of large scale turbulent combustion. In *Proc. of IEEE eScience* (2009). (Cited pages vii, 11, 131, 201, and 214.)

[BWT*11] Bremer P., Weber G., Tierny J., Pascucci V., Day M., Bell J.: Interactive exploration and analysis of large scale simulations using topology-based data segmentation. *IEEE Transactions on Visualization and Computer Graphics* (2011). (Cited pages vii, 11, 64, 131, 201, and 214.)

[BZK09] Bommes D., Zimmer H., Kobbelt L.: Mixed-integer quadrangulation. *ACM Transactions on Graphics (Proc. of ACM SIGGRAPH) 28*, 3 (2009), 1–10. (Cited page 170.)

[CCSG*09] Chazal F., Cohen-Steiner D., Guibas L. J., Memoli F., Oudot S. Y.: Gromov–hausdorff stable signatures for shapes using persistence. *Computer Graphics Forum (Proc. of SGP)* (2009). (Cited page 61.)

[CD14] Carr H., Duke D.: Joint contour nets. *IEEE Transactions on Visualization and Computer Graphics* (2014). (Cited pages 18 and 203.)

[CdCCG*14] Chaudret R., de Courcy B., Contreras-Garcia J., Gloaguen E., Zehnacker-Rentien A., Mons M., Piquemal J.-P.: Unraveling non-covalent interactions within flexible biomolecules: from electron density topology to gas phase spectroscopy. *Physical Chemistry Chemical Physics* (2014), –. (Cited pages 144, 148, and 152.)

[CGJK*11] Contreras-GarciÌa J., Johnson E. R., Keinan S., Chaudret R., Piquemal J.-P., Beratan D. N., Yang W.: Nciplot: A program for plotting noncovalent interaction regions. *J. Chem. Theo. and Comp. 7*, 3 (2011). (Cited pages 144 and 147.)

[CGT*15] Carr H., Geng Z., Tierny J., Chattopadhyay A., Knoll A.: Fiber surfaces: Generalizing isosurfaces to bivariate data. *Computer Graphics Forum (Proc. of EuroVis)* (2015). (Cited pages vii, 18, 193, 204, 205, and 215.)

[Che95a] Cheng R. K.: Velocity and scalar characteristics of premixed turbulent flames stabilized by weak swirl. *Combust. Flame 101*, 1-2 (1995), 1–14. (Cited page 133.)

[Che95b] Chernayev E.: *Marching cubes 33: Construction of topologically correct isosurfaces*. Tech. rep., CN/95-17, 1995. (Cited pages 189 and 191.)

[CLRS09] Cormen T., Leiserson C. E., Rivest R. L., Stein C.: *Introduction to Algorithms*. MIT Press, 2009. (Cited page 63.)

[CMEH*03] Cole-McLaughlin K., Edelsbrunner H., Harer J., Natarajan V., Pascucci V.: Loops in Reeb graphs of 2-manifolds. In *Proc. of ACM Symposium on Computational Geometry* (2003), pp. 344–350. (Cited pages 9, 56, 62, 76, 90, and 96.)

[Coh73] Cohen M.: *A course in simple-homotopy theory*. Springer-Verlag, 1973. (Cited page 112.)

[CR08] Chen L., Rong Y.: *Linear time recognition algorithms for topological invariants in 3D*. Tech. rep., Arxiv technical report, 2008. (Cited page 186.)

[CSA00] Carr H., Snoeyink J., Axen U.: Computing contour trees in all dimensions. In *Proc. of Symposium on Discrete Algorithms* (2000), pp. 918–926. (Cited pages 9, 62, 89, 92, 94, 95, 98, 99, and 100.)

[CSEH05] Cohen-Steiner D., Edelsbrunner H., Harer J.: Stability of persistence diagrams. In *Proc. of ACM Symposium on Computational Geometry* (2005). (Cited pages 53 and 61.)

[CSvdP04] Carr H., Snoeyink J., van de Panne M.: Simplifying flexible isosurfaces using local geometric measures. In *Proc. of IEEE VIS* (2004), pp. 497–504. (Cited pages 64, 107, and 108.)

[DA13] DOE, ASCAC: *Synergistic Challenges in Data-Intensive Science and Exascale Computing*. Tech. rep., DoE Advanced Scientific Computing Advisory Committee, Data Sub-committee, 2013. (Cited pages 15, 195, and 196.)

[DBB*09] Day M., Bell J., Bremer P.-T., Pascucci V., Beckner V., Lijewski M.: Turbulence effects on cellular burning structures in lean premixed hydrogen flames. *Combustion and Flame 156* (2009), 1035–1045. (Cited pages 141 and 142.)

[DBG*06] Dong S., Bremer P.-T., Garland M., Pascucci V., Hart J.: Spectral surface quadrangulation. *ACM Transactions on Graphics (Proc. of ACM SIGGRAPH)* (2006). (Cited page 119.)

[DFFIM15] De Floriani L., Fugacci U., Iuricich F., Magillo P.: Morse complexes for shape segmentation and homological analysis: discrete models and algorithms. *Computer Graphics Forum* (2015). (Cited page 67.)

[DG98] Dey T. K., Guha S.: Computing homology groups of simplicial complexes in $\mathbb{R}^3$. *Journal of the ACM 45* (1998), 266–287. (Cited page 90.)

[DH09] Davis T., Hager W.: Dynamic supernodes in sparse cholesky update/downdate and triangular solves. *ACM Transactions Mathematical Software 35*, 4 (2009), 1–23. (Cited page 121.)

[Dil07] Dillard S.: A contour tree library. `http://graphics.cs.ucdavis.edu/~sdillard/libtourtre/doc/html/`, 2007. (Cited pages 62 and 195.)

[DL07] Dey T., Levine J.: Delaunay meshing of isosurfaces. In *Proc. of Shape Modeling International* (2007). (Cited page 189.)

[DN08] Doraiswamy H., Natarajan V.: Efficient output sensitive construction of reeb graphs. In *International Symposium on Algorithms and Computation* (2008). (Cited pages 63, 99, 100, and 101.)

[DN13] Doraiswamy H., Natarajan V.: Computing reeb graphs as a union of contour trees. *IEEE Transactions on Visualization and Computer Graphics* (2013). (Cited page 103.)

[DSS*08] Dietrich C., Scheidegger C., Schreiner J., Comba J., Nedel L., Silva C.: Edge transformations for improving mesh quality of marching cubes. *IEEE Transactions on Visualization and Computer Graphics* (2008). (Cited page 189.)

[EH08] Edelsbrunner H., Harer J.: Persistent homology – a survey. *American Mathematical Society* (2008). (Cited page 61.)

[EH09] Edelsbrunner H., Harer J.: *Computational Topology: An Introduction*. American Mathematical Society, 2009. (Cited pages 33, 49, and 52.)

[EHNP03] Edelsbrunner H., Harer J., Natarajan V., Pascucci V.: Morse-smale complexes for piecewise linear 3-manifolds. In *Proc. of ACM Symposium on Computational Geometry* (2003). (Cited page 65.)

[EHP08] Edelsbrunner H., Harer J., Patel A. K.: Reeb spaces of piecewise linear mappings. In *Proc. of ACM Symposium on Computational Geometry* (2008). (Cited pages 17 and 203.)

[EHZ03] Edelsbrunner H., Harer J., Zomorodian A.: Hierarchical morse-smale complexes for piecewise linear 2-manfiolds. *Discrete and Computational Geometry* (2003). (Cited page 64.)

[ELZ02] Edelsbrunner H., Letscher D., Zomorodian A.: Topological persistence and simplification. *Discrete & Computational Geometry 28* (2002), 511–533. (Cited page 60.)

[EM90] Edelsbrunner H., Mucke E. P.: Simulation of simplicity: a technique to cope with degenerate cases in geometric algorithms. *ACM Transactions on Graphics 9* (1990), 66–104. (Cited pages 44, 100, and 128.)

[EMP06] Edelsbrunner H., Morozov D., Pascucci V.: Persistence-sensitive simplification of functions on 2-manifolds. In *Proc. of ACM Symposium on Computational Geometry* (2006), pp. 127–134. (Cited pages 61, 72, and 85.)

[ENS*12] Etiene T., Nonato L., Scheidegger C., Tierny J., Peters T., Pascucci V., Kirby M., Silva C.: Topology verification for isosurface extraction. *IEEE Transactions on Visualization and Computer Graphics* (2012). (Cited pages vii, 14, 163, 186, and 214.)

[ESN*09] Etiene T., Scheidegger C., Nonato L., Kirby M., Silva C.: Verifiable visualization for isosurface extraction. *IEEE Transactions on Visualization and Computer Graphics (Proc. of IEEE VIS)* (2009). (Cited page 183.)

[For01] Forman R.: A user's guide to discrete morse theory. In *Proc. of the International Conference on Formal Power Series and Algebraic Combinatorics* (2001). (Cited page 65.)

[GABCG*14] Guenther D., Alvarez-Boto R., Contreras-Garcia J., Piquemal J.-P., Tierny J.: Characterizing molecular interactions in chemical systems. *IEEE Transactions on Visualization and Computer Graphics (Proc. of IEEE VIS)* (2014). (Cited pages vii, 12, 64, 67, 131, and 214.)

[GBG*14] Gyulassy A., Bremer P., Grout R., Kolla H., Chen J., Pascucci V.: Stability of dissipation elements: A case study in combustion. *Computer Graphics Forum (Proc. of EuroVis)* (2014). (Cited page 67.)

[GBHP08] Gyulassy A., Bremer P. T., Hamann B., Pascucci V.: A practical approach to morse-smale complex computation: Scalability and generality. *IEEE Transactions on Visualization and Computer Graphics (Proc. of IEEE VIS)* (2008). (Cited pages 66 and 114.)

[GBP12] Gyulassy A., Bremer P. T., Pascucci V.: Computing morse-smale complexes with accurate geometry. *IEEE Transactions on Visualization and Computer Graphics (Proc. of IEEE VIS)* (2012). (Cited pages 66 and 114.)

[GCCG*12] Gillet N., Chaudret R., Contreras-Garcia J., Yang W., Silvi B., Piquemal J.: Coupling quantum interpretative techniques: Another look at chemical mechanisms in organic reactions. *Journal of Chemical Theory and Computation* (2012). (Cited page 146.)

[GGL*14] Gyulassy A., Guenther D., Levine J. A., Tierny J., Pascucci V.: Conforming morse-smale complexes. *IEEE Transactions on Visualization and Computer Graphics (Proc. of IEEE VIS)* (2014). (Cited pages vii, 10, 67, 105, and 214.)

[Ghr07] Ghrist R.: Barcodes: The persistent topology of data. *American Mathematical Society* (2007). (Cited page 61.)

[GND*07] Gyulassy A., Natarajan V., Duchaineau M., Pascucci V., Bringa E., Higginbotham A., Hamann B.: Topologically Clean Distance Fields. *IEEE Transactions on Visualization and Computer Graphics (Proc. of IEEE VIS) 13* (2007), 1432–1439. (Cited page 67.)

[GST14] Guenther D., Salmon J., Tierny J.: Mandatory critical points of 2D uncertain scalar fields. *Computer Graphics Forum (Proc. of EuroVis)* (2014). (Cited pages vii, 19, 193, 199, 200, 207, and 215.)

[GTJ*13] Gargouri M., Tierny J., Jolivet E., Petit P., Angelini E.: Accurate and robust shape descriptors for the identification of rib cage structures in ct-images with random forests. In *IEEE Symposium on Biomedical Imaging* (2013). (Cited page vii.)

[Gyu08] Gyulassy A.: *Combinatorial Construction of Morse-Smale Complexes for Data Analysis and Visualization*. PhD thesis, University of California at Davis, 2008. (Cited pages 58, 59, and 66.)

[Har12] Harrold W.: A journey to exascale computing. In *Proc. of SuperComputing* (2012). (Cited pages 15, 195, 196, and 197.)

[Has81] Hassin R.: Maximum flow in (s, t)-planar networks. *Information Processing Letters* (1981). (Cited page 177.)

[HSKK01] Hilaga M., Shinagawa Y., Kohmura T., Kunii T. L.: Topology matching for fully automatic similarity estimation of 3D shapes. In *Proc. of ACM SIGGRAPH* (2001). (Cited page 122.)

[HZM*08] Huang J., Zhang M., Ma J., Liu X., Kobbelt L., Bao H.: Spectral quadrangulation with orientation and alignment control. *ACM Transactions on Graphics (Proc. of ACM SIGGRAPH ASIA)* (2008). (Cited pages 119 and 170.)

[ILC*13] Ibata R., Lewis G., Conn A., Irwin M., McConnachie A., Chapman S., Collins M., Fardal M., Ferguson A., Ibata N., Mackey D., Martin N., Navarro J., Rich M., Valls-Gabaud D., Widrow L.: A vast, thin plane of corotating dwarf galaxies orbiting the andromeda galaxy. *Nature* (2013). (Cited page 1.)

[JKMS*10] Johnson E., Keinan S., Mori-Sanchez P., Contreras-Garcia J. Cohen A., Yang W.: Revealing noncovalent interactions. *Journal of the American Chemical Society* (2010). (Cited pages 144, 147, 148, and 152.)

[JP06] Joswig M., Pfetsch M. E.: Computing optimal Morse matchings. *SIAM Journal on Discrete Mathematics 20*, 1 (2006), 11–25. (Cited page 151.)

[Kit14] Kitware: Paraview cinema. `http://www.kitware.com/blog/home/post/734`, 2014. (Cited pages 10, 110, and 201.)

[KRHH11] Kasten J., Reininghaus J., Hotz I., Hege H.: Two-dimensional time-dependent vortex regions based on the acceleration magnitude. *IEEE Transactions on Visualization and Computer Graphics* (2011). (Cited page 67.)

[KTCG15] Klacansky P., Tierny J., Carr H., Geng Z.: *Fast and Exact Fiber Surfaces for Tetrahedral Meshes*. Tech. rep., University of Leeds, 2015. (Cited pages vii, 18, 193, 204, 205, and 215.)

[KVP*08] Klasky S., Vouk M., Parashar M., Khan A., Podhorszki N., Barreto R., Silver D., Parker S.: Collaborative visualization spaces for petascale simulations. In *Proc. of International Symposium on Collaborative Technologies and Systems* (2008). (Cited pages 9, 109, and 110.)

[LBM*06] Laney D. E., Bremer P., Mascarenhas A., Miller P., Pascucci V.: Understanding the structure of the turbulent mixing layer in hydrodynamic instabilities. *IEEE Transactions on*

Visualization and Computer Graphics (Proc. of IEEE VIS) (2006). (Cited page 67.)

[LC87] Lorensen W., Cline H.: Marching cubes: a high resolution 3d surface construction algorithm. In *Proc. of ACM SIGGRAPH* (1987). (Cited page 183.)

[LCGP*13] Lane J. R., Contreras-GarcÃa J., Piquemal J.-P., Miller B. J., Kjaergaard H. G.: Are bond critical points really critical for hydrogen bonding? *Journal of Chemical Theory and Computation 9*, 8 (2013), 3263–3266. (Cited pages 144, 147, 148, and 150.)

[Lew16] Lewis G.: The Atom and the Molecule. *Am. Chem. Soc.* (1916). (Cited page 144.)

[LLVT03] Lewiner T., Lopes H., Vieira A. W., Tavares G.: Efficient implementation of marching cubes cases with topological guarantees. *Journal of Graphics Tools* (2003). (Cited page 189.)

[LPG*14] Landge A., Pascucci V., Gyulassy A., Bennett J., Kolla H., Chen J., Bremer T.: In-situ feature extraction of large scale combustion simulations using segmented merge trees. In *SuperComputing* (2014). (Cited page 197.)

[MB07] Matta C., Boyd R.: *The Quantum Theory of Atoms in Molecules: From Solid State to DNA and Drug Design*. Wiley, 2007. (Cited page 148.)

[MC08] Mansour M., Chen Y.-C.: Stability characteristics and flame structure of low swirl burner. *Experimental Thermal and Fluid Science 32*, 7 (July 2008), 1390–1395. Fifth mediterranean combustion symposium. (Cited page 133.)

[MDN12] Maadasamy S., Doraiswamy H., Natarajan V.: A hybrid parallel algorithm for computing and tracking level set topology. In *International Conference on High Performance Computing* (2012). (Cited page 197.)

[Mil63] Milnor J.: *Morse Theory*. Princeton U. Press, 1963. (Cited pages 47, 77, and 186.)

[Mor34] Morse M.: *The Calculus of Variations in the Large*. No. v. 18 in Colloquium Publications - American Mathematical Society. American Mathematical Society, 1934. (Cited page v.)

[MSS94] Montani C., Scateni R., Scopigno R.: A modified look-up table for implicit disambiguation of marching cubes. *The Visual Computer* (1994). (Cited page 189.)

[MTT*13] Michelin J., Tierny J., Tupin F., Mallet C., Paparoditis N.: Quality evaluation of 3d city building models with automatic error diagnosis. In *Proc. of ISPRS Conference on SSG* (2013). (Cited page vii.)

[MW13] Morozov D., Weber G.: Distributed merge trees. In *ACM Symposium on Principles and Practice of Parallel Programming* (2013). (Cited page 197.)

[NPB*07] Nogenmyr K., Peterson P., Bai X., Nauert A., Olofsson J., Brackman C., Seyfried H., Zetterberg J. Li Z.-S., Richter M., Dreizler A., Linne M., Alden: Large eddy simulation and experiments of stratified lean premixed methane/air turbulent flames. *Proc. Combust. Inst 31* (2007), 1467–1475. *submitted to LACSEA Feature Issue*. (Cited page 133.)

[Par12] Parsa S.: A deterministic o(m log m) time algorithm for the reeb graph. In *Proc. of ACM Symposium on Computational Geometry* (2012). (Cited pages 9, 63, 103, and 213.)

[Pau60] Pauling L.: *The Nature of the Chemical Bond and the Structure of Molecules and Crystals: An Introduction to Modern Structural Chemistry*. Cornell University Press, 1960. (Cited page 144.)

[PCM03] Pascucci V., Cole-McLaughlin K.: Parallel computation of the topology of level sets. *Algorithmica* (2003). (Cited page 197.)

[POB*07] Peterson P., Olofsson J., Brackman C., Seyfried H., Zetterberg J., Richter M., Alden M., Linne M., Cheng R., Nauert A., Geyer D., Dreizler A.: Simultaneous PIV/OH PLIF, Rayleigh thermometry/OH PLIF and stereo PIV measurements in a low-swirl flame. *Appl. Opt 46* (2007), 3928–3936. (Cited page 133.)

[PP93] Pinkall U., Polthier K.: Computing discrete minimal surfaces and their conjugates. *Exp. Math.* (1993). (Cited pages 13, 120, and 165.)

[PSBM07] Pascucci V., Scorzelli G., Bremer P. T., Mascarenhas A.: Robust on-line computation of Reeb graphs: simplicity and speed. *TOG 26* (2007), 58.1–58.9. (Cited pages 62, 95, 99, 100, 101, and 108.)

[PSF08] Patane G., Spagnuolo M., Falcidieno B.: Reeb graph computation based on a minimal contouring. In *Proc. of IEEE Shape Modeling International* (2008). (Cited pages 62, 63, 71, and 87.)

[PST*13] Philip S., Summa B., Tierny J., Bremer P., Pascucci V.: Scalable seams for gigapixel panoramas. In *Eurographics Symposium on Parallel Graphics and Visualization* (2013). (Cited pages vii, 14, and 214.)

[PST*15] Philip S., Summa B., Tierny J., Bremer P., Pascucci V.: Distributed seams for gigapixel panoramas. *IEEE Transactions on Visualization and Computer Graphics* (2015). (Cited pages vii, 14, 163, 183, and 214.)

[Ree46] Reeb G.: Sur les points singuliers d'une forme de Pfaff complètement intégrable ou d'une fonction numérique. *Comptes-rendus de l'Académie des Sciences 222* (1946), 847–849. (Cited pages 9, 54, and 55.)

[RL15] Rieck B., Leitte H.: Persistent homology for the evaluation of dimensionality reduction schemes. *Computer Graphics Forum (Proc. of EuroVis)* (2015). (Cited page 61.)

[RVAL09] Ray N., Vallet B., Alonso L., Lévy B.: Geometry aware direction field design. *ACM Transactions on Graphics 29* (2009). (Cited page 170.)

[RW08] Raman S., Wenger R.: Quality isosurface mesh generation using an extended marching cubes lookup table. *Computer Graphics Forum* (2008). (Cited page 189.)

[RWS11] Robins V., Wood P., Sheppard A.: Theory and algorithms for constructing discrete morse complexes from grayscale digital images. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2011). (Cited pages 66, 112, 113, and 114.)

[SB06] Sohn B. S., Bajaj C. L.: Time varying contour topology. *IEEE Transactions on Visualization and Computer Graphics (Proc. of IEEE VIS)* (2006). (Cited page 64.)

[SCD13] Sun F., Cheng K. C., Du Q.: Modeling and segmentation of nuclei based on morse-smale complex. Submitted, 2013. (Cited page 117.)

[SENS10] Scheidegger C., Etiene T., Nonato L. G., Silva C.: *Edge flows: Stratified Morse Theory for simple, correct isosurface extraction*. Tech. rep., University of Utah, 2010. (Cited page 189.)

[SGSP15] Summa B., Gooch A., Scorzelli G., Pascucci V.: Paint and click: Unified interactions for image boundaries. *Computer Graphics Forum (Proc. of Eurographics)* (2015). (Cited page 183.)

[Shi12] Shivashankar N.: Parallel implementation of 3d morse-smale complex computation. `http://vgl.serc.iisc.ernet.in/mscomplex/`, 2012. (Cited pages 66 and 195.)

[SKK91] Shinagawa Y., Kunii T., Kergosien Y. L.: Surface coding based on morse theory. *IEEE Computer Graphics and Applications* (1991). (Cited page 62.)

[SLS07] Stelldinger P., Latecki L., Siqueira M.: Topological equivalence between a 3d object and the reconstruction of its digital image. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2007). (Cited page 186.)

[SMW06] Schaefer S., McPhail T., Warren J.: Image deformation using moving least squares. *ACM Transactions on Graphics (Proc. of ACM SIGGRAPH) 25* (2006), 533–540. (Cited page 173.)

[SN12] Shivashankar N., Natarajan V.: Parallel computation of 3d morse-smale complexes. *Computer Graphics Forum (Proc. of EuroVis)* (2012). (Cited pages 66 and 114.)

[Sou11] Sousbie T.: The persistent cosmic web and its filamentary structure: Theory and implementations. *Royal Astronomical Society* (2011). (Cited page 67.)

[SSS06] Schreiner J., Scheidegger C., Silva C.: High-quality extraction of isosurfaces from regular and irregular grids. *IEEE Transactions on Visualization and Computer Graphics* (2006). (Cited page 189.)

[STB*13] Summa B., Tierny J., Bremer P., Scorzelli G., Pascucci V.: *Active Stitching: Beyond Batch Processing of Panoramas*. Tech. rep., University of Utah, 2013. (Cited pages 179 and 183.)

[STK*09] Santos E., Tierny J., Khan A., Grimm B., Lins L., Freire J., Pascucci V., Silva C., Klasky S., Barreto R., Podhorszki N.: Enabling advanced visualization tools in a web-based simulation monitoring system. In *Proc. of IEEE eScience* (2009). (Cited pages vii, 10, 105, 109, and 214.)

[STP12] Summa B., Tierny J., Pascucci V.: Panorama weaving: Fast and flexible seam processing. *ACM Transactions on Graphics (Proc. of ACM SIGGRAPH)* (2012). (Cited pages vii, 14, 163, and 214.)

[TBTB12] Thiery J., Buchholz B., Tierny J., Boubekeur T.: Analytic curve skeletons for 3d surface modeling and processing. *Computer Graphics Forum (Proc. of Pacific Graphics)* (2012). (Cited page vii.)

[TDN*11] Tierny J., Daniels J., Nonato L., Pascucci V., Silva C.: Inspired quadrangulation. *Computer Aided Design (Proc. of ACM Solid and Physical Modeling)* (2011). (Cited pages vii, 13, 163, and 214.)

[TDN*12] Tierny J., Daniels J., Nonato L. G., Pascucci V., Silva C.: Interactive quadrangulation with Reeb atlases and connectivity textures. *IEEE Transactions on Visualization and Computer Graphics* (2012). (Cited pages vii, 11, 13, 105, 163, and 214.)

[TGP14] Tierny J., Guenther D., Pascucci V.: Optimal general simplification of scalar fields on surfaces. In *Topological and Statistical Methods for Complex Data*. Springer, 2014. (Cited page 85.)

[TGSP09] Tierny J., Gyulassy A., Simon E., Pascucci V.: Loop surgery for volumetric meshes: Reeb graphs reduced to con-

tour trees. *IEEE Transactions on Visualization and Computer Graphics (Proc. of IEEE VIS) 15* (2009), 1177–1184. (Cited pages vii, 9, 63, 69, 76, 105, and 213.)

[Tie08] Tierny J.: *Reeb graph based 3D shape modeling and applications*. PhD thesis, Lille1 University, 2008. (Cited pages 7 and 64.)

[Tie09] Tierny J.: vtkReebGraph class collection. `http://www.vtk.org/doc/nightly/html/classvtkReebGraph.html`, 2009. (Cited pages 63 and 195.)

[TN14] Thomas D. M., Natarajan V.: Multiscale symmetry detection in scalar fields by clustering contours. *IEEE Transactions on Visualization and Computer Graphics (Proc. of IEEE VIS)* (2014). (Cited page 64.)

[TP12] Tierny J., Pascucci V.: Generalized topological simplification of scalar fields on surfaces. *IEEE Transactions on Visualization and Computer Graphics (Proc. of IEEE VIS)* (2012). (Cited pages vii, 8, 13, 61, 69, 71, 163, 213, and 214.)

[TTB12] Thiery J., Tierny J., Boubekeur T.: Cager: Cage-based reverse engineering of animated 3d shapes. *Computer Graphics Forum* (2012). (Cited page vii.)

[TTB13] Thiery J., Tierny J., Boubekeur T.: Jacobians and hessians of mean value coordinates for closed triangular meshes. *The Visual Computer* (2013). (Cited page vii.)

[TV98] Tarasov S., Vyali M.: Construction of contour trees in 3d in o(n log n) steps. In *Proc. of ACM Symposium on Computational Geometry* (1998). (Cited page 62.)

[TVD06a] Tierny J., Vandeborre J.-P., Daoudi M.: 3D mesh skeleton extraction using topological and geometrical analyses. In *Proc. of Pacific Graphics* (2006). (Cited page 7.)

[TVD06b] Tierny J., Vandeborre J.-P., Daoudi M.: Invariant high level Reeb graphs of 3D polygonal meshes. In *Proc. of IEEE 3DPVT* (2006). (Cited page 7.)

[TVD07a] Tierny J., Vandeborre J.-P., Daoudi M.: Reeb chart unfolding based 3D shape signatures. In *Proc. of Eurographics* (2007). (Cited page 7.)

[TVD07b] Tierny J., Vandeborre J.-P., Daoudi M.: Topology driven 3D mesh hierarchical segmentation. In *Proc. of IEEE Shape Modeling International* (2007). (Cited page 7.)

[TVD08a] Tierny J., Vandeborre J.-P., Daoudi M.: Enhancing 3D mesh topological skeletons with discrete contour constrictions. *The Visual Computer* (2008). (Cited page 7.)

[TVD08b] Tierny J., Vandeborre J.-P., Daoudi M.: Fast and precise kinematic skeleton extraction of 3D dynamic meshes. In *Proc. of IEEE ICPR* (2008). (Cited page 7.)

[TVD09] Tierny J., Vandeborre J.-P., Daoudi M.: Partial 3D shape retrieval by Reeb pattern unfolding. *Computer Graphics Forum* (2009). (Cited page 7.)

[vGW94] van Gelder A., Wilhelms J.: Topological considerations in isosurface generation. *ACM Transactions on Graphics* (1994). (Cited page 186.)

[vKvOB*97] van Kreveld M., van Oostrum R., Bajaj C., Pasucci V., Schikore D.: Contour trees and small seed sets for isosurface traversal. In *Proc. of ACM Symposium on Computational Geometry* (1997). (Cited pages 62 and 64.)

[Wal70] Wall C. T. C.: *Surgery on compact manifolds*. American Mathematical Society, 1970. (Cited page 89.)

[WDC*07] Weber G., Dillard S. E., Carr H., Pascucci V., Hamann B.: Topology-controlled volume rendering. *IEEE Transactions on Visualization and Computer Graphics* (2007). (Cited page 64.)

[WHS*12] Wienert S., Heim D., Saeger K., Stenzinger A., Beil M., Hufnagl P., Dietel M., Denkert C., Klauschen F.: Detection and segmentation of cell nuclei in virtual microscopy images: A minimum-model approach, 2012. (Cited page 119.)

[WMKG07] Wardetzky M., Mathur S., Kalberer F., Grinspun E.: Discrete Laplace operators: no freen lunch. In *Proc. of SGP* (2007), pp. 33–37. (Cited pages 13 and 165.)

[XZCOX09] Xu K., Zhang H., Cohen-Or D., Xiong Y.: Dynamic harmonic fields for surface processing. *Comput. Graph. (Proc. of SMI) 33* (2009), 391–398. (Cited pages 121 and 165.)