Topology and Morphology of Bounded Vector Fields

Von der Fakultät Informatik, Elektrotechnik und
Informationstechnik der Universität Stuttgart
zur Erlangung der Würde eines
Doktors der Naturwissenschaften (Dr. rer. nat.)
genehmigte Abhandlung

Vorgelegt von

Gustavo Mello Machado

aus Recife, Brasilien

Hauptberichter:	Prof. Dr. Thomas Ertl
Mitberichter:	Prof. Dr. Filip Sadlo

Tag der mündlichen Prüfung: 6. März 2015

Visualisierungsinstitut
der Universität Stuttgart

2015

"The most beautiful thing we can experience is the mysterious. It is the source of all true art and science."

Albert Einstein, 1930

Acknowledgments

I am honored to have this work advised by Professor Thomas Ertl. Since our first dinner at *Churrascaria Barranco* in Porto Alegre, together with our wives, he inspired me not only with his academic competence, but also as an example of kindness. I am also deeply grateful to Professor Filip Sadlo. It was always very motivating to share his tireless desire to make science. He introduced me to a new and fascinating research field, to which I hope to continue contributing in the future.

I would like to thank Thomas Müller and Daniel Müller for the excellent collaboration in this work. Moreover, it was a pleasure to share office spaces in chronological order with Martin Falk, Marco Ament, Bertram Thomaß, Steffen Frey, and Finian Mwalongo. Special thanks to Finian, who I consider a friend for a lifetime. I thank Alexandros Panagiotidis, Bertram Thomaß, Florian Herimerl, David Körner, Finian Mwalongo, and Oliver Fernandes for the cooperation in didactic activities. I also thank all colleagues for the amazing experience I had at VIS/VISUS.

Financial support was provided by Capes-Brazil (BEX 4097/10-6), the German Research Foundation within the Cluster of Excellence in Simulation Technology (EXC 310/1) and the Collaborative Research Center SFB-TRR 75 at the University of Stuttgart.

I am thankful for the support given by my family from accross the Ocean, who assisted me motivationally and financially. Above all, I am profoundly thankful to my wife. She resigned from her activities, accompained me to Germany, and gave me a son. Sharon and Gael are my strength and joy. They remind me what really matters in life.

Contents

List of Abbreviations and Acronyms

AIA	Atmospheric Imaging Assembly
AR	Active Region
c-space	Computational Space
CC	Carrington Coordinates
CFD	Computational Fluid Dynamics
CME	Coronal Mass Ejection
CPU	Central Processing Unit
CR	Carrington Rotation
CRLN	Carrington Longitude
CRLT	Carrington Latitude
CUDA	Compute Unified Device Architecture
DOPRI5	Dormand-Prince Fifth-Order Method
ESA	European Space Agency
EVE	Extreme Ultraviolet Variability Experiment
FITS	Flexible Image Transport System
FTLE	Finite-Time Lyapunov Exponent
GLSL	OpenGL Shading Language
GONG	Global Oscillation Network Group
GPU	Graphics Processing Unit
HEK	Heliophysics Event Knowledgebase
HMI	Helioseismic and Magnetic Imager
ICME	Interplanetary Coronal Mass Ejection
IDL	Interactive Data Language
JP2	JPEG 2000
JPEG	Joint Photographic Experts Group
JPIP	JPEG 2000 Interactive Protocol
JSOC	Joint Science Operations Center
LCEA	Lambertian-Cylindrical-Equal-Area
LIC	Line Integral Convolution
LWS	Living With a Star
MAS	Magnetohydrodynamics Around a Sphere
MDI	Michelson Doppler Imager
MHD	Magnetohydrodynamics
NASA	National Aeronautics and Space Administration

NLFFF	Nonlinear Force Free Field
ODE	Ordinary Differential Equation
OpenGL	Open Graphics Library
p-space	Physical Space
PFSS	Potential-Field Source-Surface
pixel	Picture Element
RAM	Random-Access Memory
RK4	Runge-Kutta Fourth-Order Method
ROI	Region of Interest
SDO	Solar Dynamics Observatory
SH	Spherical Harmonics
SOHO	Solar and Heliospheric Observatory
SRS	Solar Region Summary
SSW	SolarSoftWare
SWPC	Space Weather Prediction Center
texel	Texture Element
voxel	Volume Element

Units

fps	Frames per second
GiB	Gibibyte
MiB	Mebibyte

List of Symbols

Symbol	**Unit**	**Meaning**
$R_\odot$		Solar radius
$R_\oplus$		Earth's radius
$\mathbf{u}$		Vector field
$\mathbf{x}(\tau)$		Tangent curve parametrization
$\mathbf{x}_h$		Hyperbolic critical element
$\mathbf{x}_0$		Hyperbolic critical point
$\mathbf{x}_c$		Hyperbolic periodic orbit
W^s		Stable manifold
W^u		Unstable manifold
λ_i		Eigenvalue
e_i		Eigenvector
$\parallel$		Parallel or antiparallel

Notation	**Meaning**
$\cdot$	Dot product
$\times$	Cross product
det	Determinant
∇	Gradient
$\nabla\cdot$	Divergence
$\nabla\times$	Curl
$\mathbf{J}$	Jacobian matrix
$\lVert\cdot\rVert$	Norm
$\lvert\cdot\rvert$	Absolute value
$\Re$	Real part
$\Im$	Imaginary part

Abstract

Vector fields are a fundamental concept in science, and as they can represent properties from electromagnetic fields to the dynamics of fluid flow, their visualization assists the study and comprehension of many physical phenomena. Aiming to extract the global structure of streamlines with respect to regions of qualitatively different behavior, the concept of vector field topology basically consists of locating singularities, i.e., critical points and periodic orbits, and computing the sets of streamlines that converge to them in positive or negative direction, called separatrices. On the one hand, this approach has proven its valuable contributions to scientific visualization, on the other hand, its limitations with respect to bounded domains have not yet been sufficiently researched, and only comparably few approaches exist for such configurations. This thesis contributes to vector field visualization, in particular to vector field topology on bounded domains, with new techniques that range from feature extraction, integration-based approaches, and topology-based approaches. More specifically, the contributions of this thesis are the following.

A local extraction technique for bifurcation lines is proposed, together with the extraction of their manifolds. Bifurcation lines represent a topological feature that has not yet been sufficiently recognized in scientific visualization. The bifurcation lines are extracted by a modification of the vortex core line extraction techniques due to Sujudi and Haimes, and Roth and Peikert, both formulated using the parallel vectors operator. While the former formulation provides acceptable results only in configurations with high hyperbolicity and low curvature of the bifurcation lines, the latter operates only well in configurations with low hyperbolicity but is able to perform well with strong curvature of the bifurcation lines, however, with the drawback that it often fails to provide a solution. The refinement of the solutions of the parallel vectors operator is presented as a means to improve both criteria and, in particular, to refine the solutions of the Sujudi and Haimes criterion in cases where the Roth and Peikert criterion fails. This technique is exemplified on synthetic data, data from computational fluid dynamics, and on magnetohydrodynamics data. As a particularly interesting application, it is demonstrated that this technique is able to extract saddle-type periodic orbits locally, and in case of high hyperbolicity at higher accuracy than traditional techniques based on integral curves.

Solar dynamics data, particularly those from the Solar Dynamics Observatory, are now available in a sheer volume that is hard to investigate with traditional visualization tools, which mainly display 2D images. While the challenge of data access and browsing has been solved by web-based interfaces and efforts like the Helioviewer project, the approaches so far only provide 2D visualizations. The visualization of such data in the full 3D context is presented, providing appropriate coordinate systems and projection techniques, including time. Methods from volume rendering and flow visualization are applied to 3D solar magnetic fields, which are derived from the sensor data in an interactive process. They are applied and extended to the space-time visualization of photospheric data, and a view-dependent visualization of coronal holes is presented. This work concentrates on two solar phenomena: the structure and dynamics of coronal loops, and the temporal evolution of the plasma convection in close vicinity of sunspots over time. This approach avoids the time coherence issue inherent in traditional magnetic field line placement, providing insight in the magnetic field and the structure of the coronal plasma. The presented techniques are also applicable in many other fields, such as terrestrial magnetospheric physics, or magnetohydrodynamics simulations.

Inspired by the view-dependent visualization of coronal holes, this thesis also presents a technique to visualize the streamline-based mapping between the boundary of a simply-connected subregion of arbitrary 3D vector fields. While the streamlines are seeded on one part of the boundary, the remaining part serves as escape border. Hence, the seeding part of the boundary represents a map of streamline behavior, indicating if streamlines reach the escape border or not. Since the resulting maps typically exhibit a very fine and complex structure and are thus not amenable to direct sampling, this approach instead aims at topologically consistent extraction of their boundary. It is shown that isocline surfaces of the projected vector field provide a robust basis for streamsurface-based extraction of these boundaries. The utility of this technique is demonstrated in the context of transport processes using vector field data from different domains like Magma flow, coronal magnetic fields for the extraction of coronal holes, and computational fluid dynamics.

Streamsurfaces are of fundamental importance to the visualization of flows. Among other features, they offer strong capabilities in revealing flow behavior (e.g., in the vicinity of vortices), and are an essential tool for the computation of 2D separatrices in vector field topology. Computing streamsurfaces is, however, typically expensive due to the difficult triangulation involved, in particular when triangle sizes are kept in the order of the size of a pixel. Different image-based approaches are here investigated for rendering streamsurfaces without triangulation, and a new technique that renders them by dense streamlines is proposed. Although this technique does not perform triangulation, it does not depend on user parametrization to avoid noticeable gaps. A GPU-based implementation shows that this technique provides interactive frame rates and low memory usage in practical applications. It is also shown that previous texture-based flow visualization approaches can be integrated with this method, for example, for the visualization of flow direction with line integral convolution.

German Abstract
—Zusammenfassung—

Vektorfelder stellen ein grundlegendes Konzept in der Wissenschaft dar, und da sie Eigenschaften in weiten Bereichen repräsentieren können, von elektromagnetischen Feldern zur Dynamik von Strömungen, unterstützt ihre Visualisierung die Untersuchung und das Verständnis vieler physikalischer Phänomene. Mit dem Ziel, die globale Struktur von Stromlinien bezüglich der Regionen ihres qualitativ ähnlichen Verhaltens zu extrahieren, besteht das Konzept der Vektorfeld-Topologie grundsätzlich darin, Singularitäten, bestehend aus kritischen Punkten und periodischen Orbits, zu bestimmen und die Mengen von Stromlinien, Separatrizen genannt, welche gegen diese Singularitäten vorwärts oder rückwärts konvergieren, zu berechnen. Auf der einen Seite hat sich die Vektorfeld-Topologie in der wissenschaftlichen Visualisierung bewährt, andererseits sind ihre Limitierungen hinsichtlich der Anwendbarkeit auf begrenzten räumlichen Gebieten noch nicht ausreichend erforscht und es stehen nur vergleichsweise wenige Ansätze für solche Konfigurationen zur Verfügung. In dieser Arbeit wurden neue Techniken auf dem Gebiet der Vektorfeld-Visualisierung entwickelt, insbesondere der Vektorfeld-Topologie auf begrenzten Gebieten, mit neuen Techniken, welche sich von der Merkmalsextraktion über integrationsbasierte Ansätze zu topologischen Methoden erstrecken. Insbesondere liefert diese Dissertation folgende Beiträge.

Es wird eine Extraktionstechnik für Bifurkationslinien und deren Mannigfaltigkeiten vorgestellt. Bifurkationslinien stellen ein topologisches Merkmal dar, welches in der wissenschaftlichen Visualisierung noch nicht ausreichend Beachtung gefunden hat. Die Bifurkationslinien werden mittels einer Modifikation der Techniken zur Wirbelkernextraktion von Sujudi und Haimes sowie Roth und Peikert extrahiert, in der Formulierung des Parallel Vectors Operators. Während die erstgenannte Technik nur in Fällen mit starker Hyperbolizität und schwacher Krümmung der Bifurkationslinien akzeptable Resultate liefert, funktioniert die zweite nur gut in Konfigurationen mit schwacher Hyperbolizität, kann dafür aber auch stark gekrümmte Bifurkationslinien extrahieren. Andererseits gelingt es der zweiten Methode oft gar nicht, eine Bifurkationslinie zu extrahieren. Diese Schwächen motivieren unseren Ansatz, die Lösungskurven des Parallel Vectors Operators beider Ansätze zu verfeinern, insbesondere die des Ansatzes von Sujudi und Haimes, in Fällen, in denen die Methode von Roth und Peikert keine Lösung liefert. Diese Technik wird mittels synthetischer Daten, Daten aus der numerischen Strömungsmechanik und magnetohydrodynamischer Daten demonstriert und evaluiert. Als eine besonders interessante Anwendung wird gezeigt, dass mittels dieser lokalen Technik periodische Orbits vom Typ Sattel extrahiert werden können und in Fällen hoher Hyperbolizität mit höherer Genauigkeit als mittels traditioneller Techniken basierend auf Integralkurven.

Daten der Solardynamik, insbesondere die vom Solar Dynamics Observatory, sind heutzutage in einem Umfang verfügbar, dass es schwierig ist, diese mittels traditioneller Techniken zu untersuchen, welche sich hauptsächlich auf die Darstellung zweidimensionaler Bilder beschränken. Während die Herausforderung des Datenzugriffs und der Suche in diesen Daten mittels Web-Schnittstellen und Ansätzen wie dem Helioviewer-Projekt bereits gelöst sind, liefern diese Ansätze bisher nur zweidimensionale Visualisierungen. In dieser Dissertation wird die Visualisierung solcher Daten im vollen 3D Kontext vorgestellt, basierend auf geeigneten Koordinatensystemen und Projektionstechniken, wobei Zeit explizit einbezogen wird. Es werden Methoden der Volumenvisualisierung und Strömungsvisualisierung auf dreidimensionale solare Magnetfelder angewandt, welche aus Sensordaten in einem interaktiven Prozess abgeleitet werden. Diese Methoden werden zur räumlich-zeitlichen Visualisierung photosphärischer Daten angewandt und erweitert, und es werden betrachterabhängige Visualisierungstechniken für koronale Löcher vorgestellt. Diese Arbeit fokussiert auf zwei solare Phänomene: die Struktur und Dynamik koronaler Löcher und die zeitliche Entwicklung der Konvektion des Plasmas in der Nähe von Sonnenflecken. Der hierfür vorgestellte Ansatz vermeidet die mit traditionellen Ansätzen verbundenen Probleme der zeitlichen Kohärenz und erlaubt Einblicke in das Magnetfeld und die Struktur des koronalen Plasmas. Die vorgestellten Techniken können in vielen weiteren Gebieten eingesetzt werden, beispielsweise der Physik der Magnetospähre der Erde und der Magnetohydrodynamik.

Inspiriert durch die betrachterabhängige Visualisierungstechnik für koronale Löcher, präsentiert diese Dissertation auch eine Technik, um die stromlinienbasierte Abbildung zwischen dem Rand eines einfach zusammenhängenden Teilbereichs beliebiger 3D Vektorfelder zu visualisieren. Während die Stromlinien auf einem Teil des Randes gesät werden, dient der restliche Rand als sogenannte Flucht-Schwelle. Somit stellt der Saatbereich eine Abbildung des Stromlinienverhaltens dar und zeigt an, ob die an einem Punkt gestartete Stromlinie die Flucht-Schwelle erreicht oder nicht. Da die resultierenden Abbildungen typischerweise eine sehr feine und komplexe Struktur aufweisen und deshalb nicht mittels direkter Abtastung zugänglich sind, hat der vorgestellte Ansatz stattdessen die Extraktion ihrer Ränder zum Ziel. Es wird gezeigt, dass Isoklinflächen des projizierten Vektorfeldes eine robuste Basis für eine Stromflächen-basierte Extraktion darstellen. Der Nutzen dieser Technik wird im Kontext von Transportprozessen demonstriert, anhand Vektorfeld-Daten aus verschiedenen Gebieten, wie zum Beispiel der Magmaströmung, der koronalen Magnetfelder zur Extraktion koronaler Löcher und der numerischen Strömungsmechanik.

Stromflächen haben eine grundlegende Bedeutung in der Visualisierung von Strömungen. Neben anderen Merkmalen, sind sie besonders nützlich, um Strömungseigenschaften (z.B. in der Nähe von Wirbeln) aufzuzeigen und stellen ein essenzielles Werkzeug dar für die Extraktion zweidimensionaler Separatrizen in der Vektorfeld-Topologie. Die Berechnung von Stromflächen ist, nichtdestotrotz, ein aufwändiges Unterfangen aufgrund der involvierten schwierigen Triangulierung, insbesondere wenn die Größe der Dreiecke in der Größenordnung der Pixelgröße liegt. Verschiedene bildbasierte Ansätze zur Stromflächendarstellung ohne Triangulierung werden hier untersucht, und es wird eine Technik vorgestellt, welche diese mittels dicht gesäter Stromlinien darstellt. Obwohl diese Technik keine Triangulierung erzeugt, benötigt sie keine Parametrisierung durch den Benutzer, um eine lückenlose Darstellung sicherzustellen. Eine GPU-basierte Implementierung zeigt, dass diese Technik interaktive Bildwiederholraten bietet und in typischen Anwendungen geringe Speicherplatzanforderungen aufweist. Es wird auch gezeigt, dass bestehende Ansätze zur texturbasierten Strömungsvisualisierung mit dieser Technik integriert werden können, beispielsweise, um die Strömungsrichtung mittels Linienintegral-Faltung darzustellen.

Chapter 1

Introduction

Vector fields are a concept at the core of science and engineering and they are defined as the assignment of magnitude and direction at each point in space and time. As such, they can represent, for example, velocity, force, or acceleration. Practical examples of vector fields are the fluid flow dynamics of liquids and gases, the electromagnetic fields of charged particles, the Earth's magnetosphere, the Coronal magnetic field, and gravitational fields. Since they represent a directional property in general, one can visualize that property locally with glyphs. Figure 1.1a shows the visualization of a two-dimensional (2D) electric field with glyphs as arrows. The orientation of arrows illustrates the field's direction. Note that, the arrows are often also colored or stretched to represent magnitude. Vector fields, however, typically represent transport along directions and their analysis usually needs to take into account their global structure, which is reflected by integral curves, such as streamlines. While there is a large body of literature on integration-based visualization [Hultquist, 1992; Turk and Banks, 1996; Zöckler et al., 1996; Jobard and Lefer, 1997; McLoughlin et al., 2010], there is a subfield, denoted vector field topology, which aims at the extraction of the global structure of streamlines [Perry and Chong, 1987; Helman and Hesselink, 1989; Globus et al., 1991; Asimov, 1993]. In analogy to geometric topology, one analyzes the connectivity of the domain by means of streamlines, i.e., regions that exhibit coherent streamline shape represent connected regions and are visualized as such. To do so, the boundaries between these regions are of particular interest—they represent so-called separatrices that separate the qualitatively different regions, and they represent manifolds of streamlines, often *streamsurfaces*, that converge in forward or reverse direction to saddle-type critical points, which are isolated zeros of the vector field, or to periodic orbits, which are isolated closed streamlines. Hence, to decide if a quantity can be transported by a vector field between two points, it is sufficient in unbounded domains to test if these two points reside in the same region of coherent behavior. Figures 1.1 (b) and (c) compare the visualization of an electric field with (b) Line Integral Convolution (LIC) [Cabral and Leedom, 1993], which is an established integration-based technique to visualize 2D vector fields that samples an input noise texture along integral curves to produce a "flow pattern" as an output image, and (c) LIC combined with traditional vector field topology.

An impeding difficulty with the traditional vector field topology, however, arises if the domain is bounded like in typical discretized datasets, or if the transport in subregions is of interest, for example, to locally analyze large and complex datasets. In such cases, concepts from traditional vector field topology cannot provide the answer in generic configurations because the required critical points, periodic orbits, and separatrices can be at least partially located outside the domain and are hence not available. The topology of bounded vector fields is handled by a few previous works [Helman and Hesselink, 1991; Kenwright et al., 1999; Scheuermann et al., 2003; Weinkauf et al., 2004]. These techniques typically detect specific flow features at the boundaries to complement the traditional vector field topology (Figure 1.1d). However, this can be insufficient for specific cases because it does not consider the flow behavior in the interior of the domain boundaries.

A line feature called *bifurcation line* is closely related to the concept of vortex core lines [Sujudi and Haimes, 1995; Roth and Peikert, 1998] with the difference that instead of exhibiting vorticity along it, which relates to rotational behavior in its vicinity, it exhibits hyperbolicity, i.e., expanding and contracting flow behavior. Bifurcation lines are not yet sufficiently investigated by the scientific visualization community [Perry and Chong, 1987; Roth, 2000] and they can be thought of as analogous of the separation and attachment lines [Kenwright et al., 1999]. Separation lines are the loci at boundaries where the vector field while flowing in the direction of a boundary splits due to the contact with that boundary, and attachment lines are the reverse analogous. Bifurcation lines are known to provide local topological insights [Perry and Chong, 1987] because they present two manifolds of streamlines, one converging to them in forward and one in reverse time, which may or may not contribute to the global vector field topology, depending on the feature strength or connectivity, e.g., with other features. Bifurcation lines are a further tool to analyze the topology of vector fields in the absence of critical points and periodic orbits, because they

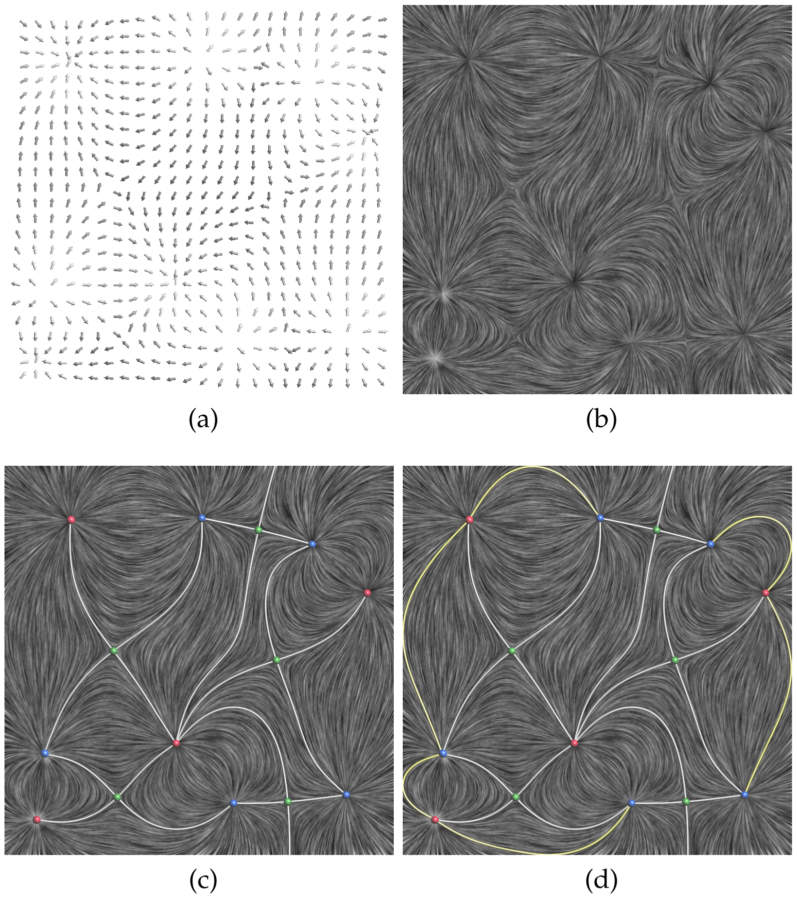


Figure 1.1 — 2D electric field generated by a set of charged particles. (a) Direct visualization with glyphs as arrows that, in this example, shows only the local field direction by the orientation of the arrows. Integration-based techniques provide global representation of the vector field and more accurate perception of the transport, e.g., (b) with LIC. (c) combines LIC with vector field topology, which finds saddle-type critical points (green points) and computes their separatrices (white curves) to extract the global structure of the field by means of regions with coherent streamline behavior. The critical points of types source and sink are colored red and blue, respectively. (d) Topology with respect to domain boundaries [Scheuermann et al., 2003] complements traditional vector field topology in subregions by computing separatrices (yellow curves) from the loci at the region's boundaries that exhibit switch from outflow to inflow behavior.

can provide topological insights, for example, when they reach the domain boundaries or if they are (part of) periodic orbits.

All of the aforementioned methods and features can, however, fail to provide correct solutions in specific topology-related applications, such as the computation of coronal holes, which are regions at the surface of the Sun (i.e., the photosphere) that exhibit magnetic field lines that escape into space. Coronal holes consist of a typical bounded vector field topology problem, as the subject of analysis is just the transport between boundaries from a given subregion. In accordance with astrophysics, this subregion is defined between the photosphere (i.e., the solar radius $R_\odot$) and the escape radius $R_e = 2.5R_\odot$, because the solar magnetic field is known to exhibit very low variation beyond $2.5R_\odot$. Furthermore, this kind of problem can be extended to any subregions in any vector field, which corresponds to the extraction of *escape maps*.

1.1 Contributions and Structure of the Thesis

This thesis presents novel techniques to the field of vector field visualization, vector field topology, and visualization in astrophysics. The techniques and applications presented in this thesis were published in the proceedings of conferences and in one journal on visualization [Machado et al., 2012, 2013, 2014b,a]. Chapter 2 provides background to the reader in the field of visualization of vector fields and vector field topology by reviewing concepts and introducing the notation used throughout this thesis. It also reviews the state of the art and some works related to the projects presented here. The following chapters are then structured into four projects.

First, Chapter 3 presents a technique for local extraction of bifurcation lines, which is inspired on insights from the thesis of Roth [2000]. This approach modifies the criteria for the extraction of vortex core lines according to Sujudi and Haimes [1995] and Roth and Peikert [1998], in analogy to the criteria for attachment and separation lines of Kenwright et al. [1999]. Furthermore, as these techniques exhibit inaccuracy (e.g., while extracting curved features), this chapter presents an algorithm that slightly deforms and moves the extracted line such that it approximates to a streamline as close as possible, conforming with the solution of the parallel vectors operator. It also presents a method for the extraction of the two 2D manifolds from the extracted bifurcation lines. Moreover, it investigates the proposed criteria and makes an analysis of the relation between bifurcation lines with its manifolds and other vector field topology features, like saddle-type periodic orbits [Asimov, 1993] and saddle connectors [Theisel et al., 2003].

Chapter 4 presents a project that introduces several visualization techniques to the field of solar astrophysics, including volume rendering for solar data by means of raycasting in spherical grids; 3D LIC as an alternative for field line placement in 3D magnetic fields; a data-driven LIC-based technique for visualizing the solar magnetic field that imitates the self-illustrating phenomenon of coronal loops; view-dependent and space-time visualization of photosphere imaging data and view-dependent visualization of coronal holes, which inspired the escape maps project in the next chapter.

The traditional approach from solar astrophysics for the identification of coronal holes seeds magnetic streamlines at the photosphere and tests if they reach the escape boundary at $2.5R_{\odot}$. However, this approach suffers from sampling issues with respect to the thin structures at the photosphere, as they would easily miss those structures. Moreover, although coronal holes represent solar phenomena closely related to bounded vector field topology, because they reflect the transport due to solar magnetic field lines between two boundaries of the dataset (i.e., the photosphere and the escape radius), existing vector field topology approaches cannot guarantee topologically correct extraction. Chapter 5 reformulates the problem of coronal hole extraction to vector fields in general, by identifying regions of a subset of the domain boundaries that exhibit streamlines that "escape" from the domain through other subsets of the boundaries. Furthermore, it presents a technique based on isocline surfaces that provides topologically correct solution of escape maps in any simply-connected subregion. Moreover, it introduces further applications of escape maps to fields other than solar magnetic fields, for example, to Computational Fluid Dynamics (CFD) and Earth's magma flow.

The last project (Chapter 6) presents a new technique for direct visualization of streamsurfaces, which are a fundamental tool for the visualization of vector fields in general, due to its capabilities in revealing flow structure, but also at the core of vector field topology, as they represent 2D separatrices, e.g., from critical points, periodic orbits, boundary switch curves, and bifurcation lines. This approach avoids the expensive connectivity construction involved in most of the previous approaches [Hultquist, 1992; van Wijk, 1993; Garth et al., 2008; Krishnan et al., 2009; Stöter et al., 2012; Schulze et al., 2012] and does not demand user parametrization to avoid gaps, in contrast to, e.g., the point-based approach from Schafhitzel et al. [2007]. This is achieved by an image-based approach that draws only lines and is capable of rendering with correct surface shading. Moreover, it can be complemented by previous approaches for texture advection on surfaces [Weiskopf and Ertl, 2004].

Chapter 7 concludes this thesis by outlooking its contributions and presenting some possibilities of future work.

CHAPTER 2

Visualization of Vector Fields

This chapter addresses concepts and notations of visualization of vector fields and provides an overview of the state of the art, which serves as background for the projects presented in the following chapters.

2.1 Vector Fields

A vector field can be described by

$$\mathbf{u} : \mathbb{R}^{m+k} \rightarrow \mathbb{R}^{n} \quad | \quad m, n > 0 \text{ and } k \in \{0, 1\}, \tag{2.1}$$

that maps spatio-temporal points $\mathbb{R}^{m+k}$ to vectors $\mathbb{R}^{n}$. Thus, $\mathbb{R}^{m}$ consists of an Euclidean representation of space with m dimensions, $\mathbb{R}^{k}$ is the temporal dimension, and $\mathbb{R}^{n}$ represents the field of vectors with n components. Although there are many possible configurations for this representation, the scope of this thesis is limited to 2D and 3D vector fields in which the vectors exhibit as many components as spatial dimensions, i.e., $m = n = 3$ and $m = n = 2$.

Vector fields are classified as *steady* and *unsteady* fields with respect to their dependence on time. Unsteady vector fields vary with time and are, therefore, also called time-dependent. Hence, they consist of those fields where $k = 1$ according to Equation 2.1. Steady vector fields, on the other hand, are time-independent and consist of those fields where $k = 0$. As such, steady vector fields can represent, for example, time-stamps of unsteady vector fields or vector fields of phenomena that do not vary over time.

Lipschitz Continuity

In this thesis, Lipschitz continuity of $\mathbf{u}$ is assumed, which is given by the existence of a constant $K \geq 0$ such that

$$\|\mathbf{u}(\mathbf{x}) - \mathbf{u}(\mathbf{x}')\| \leq K\|\mathbf{x} - \mathbf{x}'\| \tag{2.2}$$

for any $\mathbf{x} \neq \mathbf{x}'$. Lipschitz continuity, which consists of a property of functions classified between continuity C^0 and differential continuity C^1, is fundamental in discrete representations of fields, often found in computerized datasets, because the piecewise linear interpolation forms Lipschitz continuous functions. Thus, vector fields sampled on a cellwise manner with bi- or trilinear interpolations, as the ones found in the projects presented here, are Lipschitz continuous, as well.

Divergence

Divergence is a vector operator, denoted $\nabla\cdot$, that measures the outflow of infinitesimally small subvolumes of a vector field $\mathbf{u}$. *Divergence-free* vector fields, i.e., $\nabla \cdot \mathbf{u} = 0$, is a property encountered, e.g., in magnetic fields, which are one of the main subject of study in this thesis, and in velocity fields of incompressible fluids.

2.2 Glyphs

The most direct approach for the visualization of vector fields is with *glyphs*. Glyphs are very common in information visualization and they basically consist of a code defined by the shape, size, orientation, and color, positioned in space, typically showing local information. Probably the most straightforward glyph to represent vectors are arrow glyphs. The visualization with arrow glyphs seeded on regular grids, for example, in Figure 1.1a, is called Hedgehog, due to the pattern produced. Nevertheless, there are also more elaborate approaches that improve the visualization by defining better positioning or dedicated shape of glyphs, for example, to show the velocity gradient [de Leeuw and van Wijk, 1993] or how vectors vary over time [Hlawatsch et al., 2014]. Although, 3D visualization with glyphs often suffers from clutter and occlusion, it consists of a reasonable approach for 2D visualizations and, due to its simplicity, it is usually a nice start point for implementations of visualization applications. Laidlaw et al. [2005] performed a user study, investigating, among others, some glyph-based approaches for the visualization of 2D vector fields. Also in 3D fields, glyphs are robust tools to represent isolated features. For example, Globus et al. [1991] and Theisel et al. [2003] defined glyphs to discriminate the types of critical points in flow fields. A simple widely used approach is shown in Figure 1.1c, where color-coded sphere-shaped glyphs are used to represent the types of critical points.

2.3 Integral Curves

Integral curves, also called field lines or characteristic curves, are an elementary tool to any integration-based flow visualization, which are often preferred for the representation of the global structure of vector fields. The most fundamental integral curves are the *tangent curves*, which consist of curves that are tangent to the vector field everywhere and can represent, for example, the trajectory of massless particles in fluid flows. Tangent curves are further classified as *streamlines* and *pathlines*, to denote particle paths in steady and unsteady vector fields, respectively. Hence, given the vector field **u** and the following parametrization of tangent curves $\mathbf{x}(\tau) \in \mathbb{R}^m$, where for unsteady fields τ is time, then pathlines are defined by the following Ordinary Differential Equation (ODE):

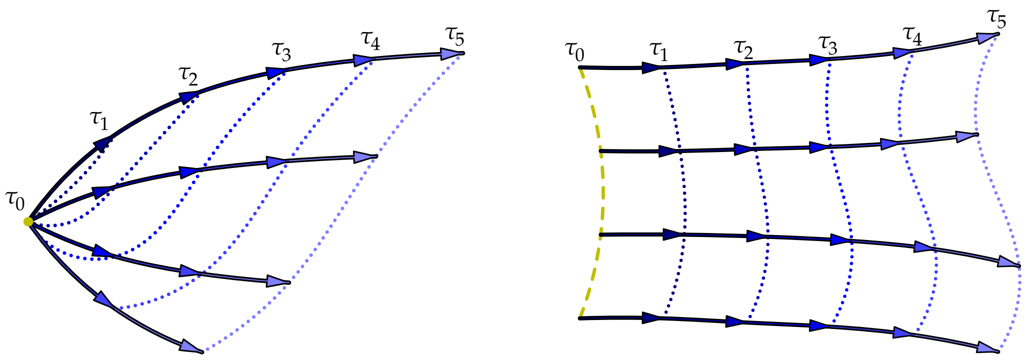
$$\frac{\partial \mathbf{x}(\tau)}{\partial \tau} = \mathbf{u}(\mathbf{x}(\tau), \tau). \tag{2.3}$$

Streamlines by the following autonomous ODE:

$$\frac{\partial \mathbf{x}(\tau)}{\partial \tau} = \mathbf{u}(\mathbf{x}(\tau), t), \quad \text{with } t = \textit{constant}. \tag{2.4}$$

Uniqueness is an interesting property of streamlines and it states that, assuming at least Lipschitz continuity, for any point **x** in a given steady vector field, there is one and only one streamline passing through **x**. In other words, streamlines cannot intersect each other. Moreover, the computation of tangent curves is employed due to an initial condition $\mathbf{x}(\tau_0)$, which characterizes the overall problem as an initial value problem. In vector field visualization, this initial condition is called *seed*.

Besides tangent curves, there are other two types of integral curves: *streaklines* and *timelines*. In analogy to pathlines, which are defined by seed points in space-time, streaklines and timelines are defined by seed curves in space-time. The seed curves of streaklines consist of a single position in space but not in time, in other words, they are seed points that remain fixed over time, while the seed curves of timelines consist of a single point in time but not in space. Thus, streaklines are curves at a given point in time, which are formed by the connection of all particles that passed through a seed point in space. They are often described in analogy to the observation of dye injection in fluid flows. Timelines are curves at a given point in time, which are formed by the



(a) (b)

Figure 2.1 — Illustrations of streaklines (a) and timelines (b). The pathlines are shown as blue curves with arrows, that are shaded from dark blue to light blue to illustrate the time increasing from τ_0 to τ_5. In (a), dotted blue curves are streaklines at different time stamps, defined by the seed point (yellow). To form the streaklines, fronts of pathlines that are released from that seed point over time $\tau_0 < \tau_i < \tau_5$ are used. While in (b), dotted blue curves are timelines defined by the seed curve (yellow). Note that, different from streaklines, to compute timelines all pathlines start at the same time $\tau = \tau_0$.

connection of all particles that passed through a given seed curve in space at a given time in the past. Note that, in steady vector fields timelines differ from the other integral curves, since pathlines and streaklines degenerate to streamlines, whereas timelines do not.

Figure 2.1 illustrates the relation between pathlines, streaklines, and timelines. In these illustrations, blue curves with arrows represent pathlines, and the shading (from dark to light blue) represents the time τ_i along the pathlines. The dotted curves in (a) are streaklines and in (b) are timelines. To compute streaklines one needs to start successive pathlines at the same position (yellow) and at different times, while to compute timelines one needs to start pathlines at different positions along the seed curve (yellow) but at the same time τ_0.

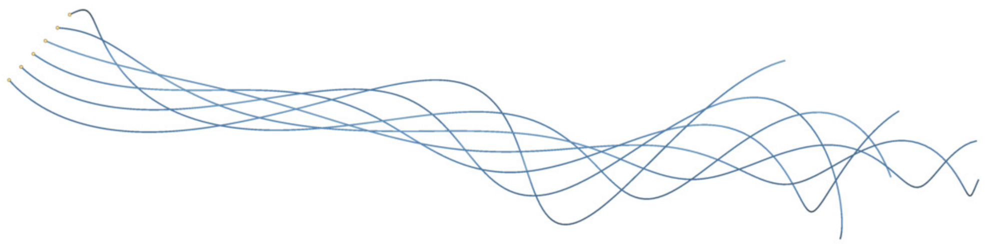
One of the most representative fields of research on the visualization of integral curves is concerned with their placement. The challenge in integral curve placement consists of finding an appropriate distribution of integral curves in order to favor the visualization. The straightforward approaches for streamline placement consist of seeding them either at the vertices of regular grids, randomly in space, or at user-defined positions. For 2D flows, Turk and Banks [1996] proposed to iteratively optimize an initial set of random streamlines to achieve uniform distribution in space, and Jobard and Lefer [1997] improved performance by selecting streamlines one by one at user-defined distance from the previously positioned streamlines. Mebarki et al. [2005] also improved the performance of streamline placement in 2D by choosing the furthest point from previously positioned streamlines to seed other ones, which allows for some longer streamlines. More recently, Wu et al. [2010] find longest orthogonal curves inside topological subregions that are used as seeding structure for evenly-spaced streamlines and provide a visualization that is coherent with topological features. Li et al. [2008], instead of finding even distributions of streamlines, propose the visualization of as few as possible streamlines without missing important features. Due to severe occlusion, there are not many works on placement of time-dependent or 3D integral curves. Li and Shen [2007] presented streamline placement in 3D that refers to their projection in image-space. Günther et al. [2011] proposed a view-dependent approach that applies transparency to specific parts of integral curves in order to reduce occlusion of relevant features in 3D. Moreover, some works investigate the placement of integral curves on surfaces [Spencer et al., 2009]. Beyond the placement of curves, there are works that addressed the rendering of streamlines, for example, to improve depth perception, to show local field behavior along curves, or for aesthetics reasons. For example, Zöckler et al. [1996] presented illuminated streamlines to enhance 3D perception, which was improved by Mallo et al. [2005].

Streamline integration is an essential step in this thesis. It is required to obtain the 1D and 2D manifolds in vector field topology (Chapters 3 and 5), to construct image-based streamsurface (Chapter 6), to integrate magnetic field lines and LIC (Chapter 4), as well in further steps of the algorithms presented in this thesis. Hence, a fast implementation is required, which we realized on the Graphics Processing Unit (GPU) using the standard fourth-order Runge-Kutta scheme. The scheme is applied in computational space to exploit, e.g., the topology of structured grids (Chapters 4 and 5) and, more important, to ease the implementation and to improve the efficiency on the GPU.

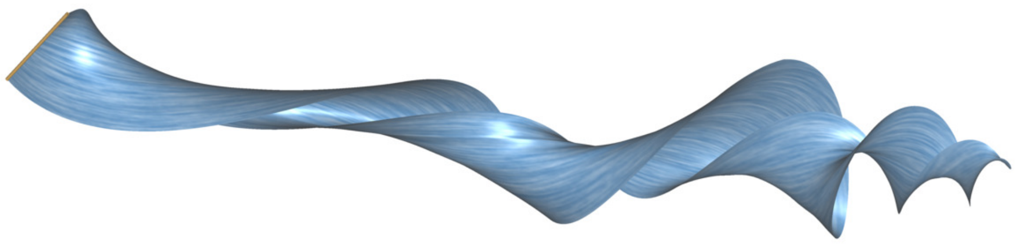
2.3.1 Integral Surfaces

The natural extension of integral curves are integral surfaces, which are likewise classified as streamsurfaces, pathsurfaces, streaksurfaces, and timesurfaces. Integral surfaces are infinite sets of integral curves that were seeded contiguously, i.e., streamsurfaces, pathsurfaces, and streaksurfaces are infinite sets of streamlines, pathlines, and streaklines seeded along curves, and timesurfaces are infinite sets of timelines seeded on surfaces.

Integral surfaces are an important tool to visualize 3D flow structure. Figure 2.2 shows a comparison between visualization by a set of streamlines (a) and a



(a)



(b)

Figure 2.2 — Comparative visualizations of vortical flow behavior with (a) streamlines, and (b) a streamsurface rendered with dense streamlines according to the image-based technique presented in Chapter 6. The seeds (yellow) on the left define the sparse streamlines and streamsurface (blue). LIC assists in the visualization of intrinsic flow behavior of streamsurfaces (b). In the experiments, the surface was rendered at 7.5 fps for a resolution of 900 × 900 pixels.

streamsurface (b), as the yellow seed points in (a) are positioned along the yellow seed curve in (b) for the same dataset. The streamsurface provides a better representation of the swirl structure, e.g., by improved depth perception.

Existing techniques for the computation of integral surfaces can be classified into *explicit* and *implicit* methods. In the former, the surfaces are computed by performing explicit integration from the seeds along the vector field. Geometric representations (usually triangle meshes) that cover the traveled area are typically used for further rendering.

The seminal technique to the explicit computation of streamsurfaces is the one presented by Hultquist [1992]. It builds triangulated streamsurfaces by performing front advancement, which is held balanced to keep the front, as much as possible, orthogonal to the velocity field. To control front density, Hultquist's approach inserts and removes streamline samples at the front to handle flow divergence and convergence, respectively. To perform insertion and removal of samples, the connectivity between the streamlines along a front is maintained by linked lists. This approach also detects internal flow deviations, which can cause very high divergence (e.g., when consecutive samples head in almost opposite directions), and splits the front at that point. Although this technique is straightforward, it faces some difficulties in generating well-discretized meshes efficiently for complex flow scenarios.

Aiming to overcome the limitations of Hultquist's technique, some further explicit approaches were proposed [Garth et al., 2008; Krishnan et al., 2009; Schulze et al., 2012]. Garth et al. [2008] divided the method into two stages to obtain accurate streamsurfaces as well as other types of integral surfaces (i.e., pathsurfaces, streaksurfaces, and timesurfaces). First, they compute a surface approximation by a series of accurate time lines, and later they use these constructs to generate the triangulation for rendering. Krishnan et al. [2009] also decouple streamline integration from mesh generation to profit from systems with multiple processors like clusters on the generation of streaksurfaces and timesurfaces. McLoughlin et al. [2009] compute quad-based streamsurfaces and pathsurfaces by adapting the speed of front advancement while detecting rotational flow behavior. Although this approach tends to perform well for non-complex tangent surfaces (e.g., without strong shear behavior), it can only be used with lower-order integrators like the Euler method, and is not suitable with higher-order integrators, which are more accurate. Recently, Schulze et al. [2012] presented a technique that scales the vector field, such that the fronts are kept as much as possible orthogonal to the flow field. This approach generates well-spaced meshes by preventing small front advancements.

In order to avoid the expensive and difficult triangulation involved in most of the techniques, Schafhitzel et al. [2007] presented a point-based approach, which is the work most closely related to the technique presented in Chapter 6. They sample a dense set of points along the streamsurfaces and apply splatting to avoid noticeable gaps on the visualization when rendering the surfaces. Their technique depends on user parametrization to prevent gaps and stores the full set of particles in a two-dimensional data structure, which, however, limits scalability. Chapter 6 presents an explicit streamsurface technique that guarantees visualization of streamsurfaces without gaps and that does not demand user parametrization for such. Moreover, it only keeps samples along the two last steps of each front during surface integration, which reduces memory usage. Interactive frame rates were achieved with a Compute Unified Device Architecture (CUDA) based implementation (see Figure 2.2).

Introduced by van Wijk [1993], implicit streamsurfaces are computed by generating a 3D scalar field that corresponds to the advection of a closed seed curves. The streamsurfaces can hence be obtained from the scalar field with any isosurface extraction technique like marching cubes [Lorensen and Cline, 1987]. Stöter et al. [2012] extended this approach by using four-dimensional representations, which allows for the computation of other types of integral surfaces in a unified framework.

McLoughlin et al. [2010] present a concise overview of many integration-based techniques including, among others, integral curves, surfaces, and volumes.

2.4 Line Integral Convolution

Cabral and Leedom [1993] presented line integral convolution (LIC) as an alternative to streamline placement. The intuitive idea behind this is that for each point p in space an input noise is convolved along the streamline defined by p to determine a scalar value for p, which reveals the flow pattern (see Figure 1.1b). Thus, LIC produces a dense output ρ, which is given by

$$\rho(p) = \int_{\tau_0 - L}^{\tau_0 + L} k(\tau - \tau_0) N(\mathbf{x}(\tau)) d\tau, \tag{2.5}$$

that evaluates ρ at the point p, where p is the seed point of the streamline (i.e., $\mathbf{x}(\tau_0) = p$), k is the convolution kernel, N is the input noise, and L defines the line integration length.

LIC is an established integration-based visualization technique for 2D steady vector fields. Further works addressed solutions for some of its limitations, for example, to show field orientation [Wegenkittl et al., 1996], to improve performance [Stalling and Hege, 1995; Zöckler et al., 1996], for unsteady fields [Shen and Kao, 1997; Sundquist, 2003], and 3D fields [Interrante and Grosch, 1997, 1998; Rezk-Salama et al., 1999]. Moreover, texture-based flow visualization on surfaces in 3D [Wijk, 2003; Laramee et al., 2003; Weiskopf and Ertl, 2004] is an important tool for the visualization of internal flow properties (see Figure 2.2b). Weiskopf and Ertl [2004], for example, presented a hybrid physical/device space representation that allows for achieving texture advection on surfaces and features with frame-to-frame coherence.

In this thesis, the hybrid physical/device space approach of Weiskopf and Ertl [2004] was implemented to demonstrate its suitability with the image-based streamsurface technique presented in Chapter 6. In Chapter 4, 3D LIC was introduced to analysis in solar physics as an alternative to field lines and was also used as a building block for a novel technique that imitates the solar phenomenon of coronal loops.

Previous to LIC, van Wijk [1991] presented spot noise, which also represents a texture-based technique that blends together random samples of so-called spot functions that are deformed due to the vector field. de Leeuw and van Liere [1998] performed a comparative study of LIC and spot noise, which concluded that spot noise is better at showing field magnitude and LIC at showing direction. Laramee et al. [2004] wrote an interesting survey, which is here recommended as further reading for those aiming an overview of texture-based techniques for the visualization of vector fields.

2.5 Vector Field Topology

Vector field topology, introduced to scientific visualization by the works due to Perry and Chong [1987], Helman and Hesselink [1989, 1991], and Globus et al. [1991], consists of the extraction of the global structure of streamlines from vector fields, and therefore, it provides a separation of the vector field into regions of qualitatively different streamline behavior. This separation is given by singularities at which the flow splits, whether in forward or in reverse time direction. This split due to hyperbolic points, thermed saddles, characterizes different behavior to the streamlines that are located at either side of the split, because due to this separation, they typically exhibit different structure and extent. Since streamlines are at the core of this method, the subject of analysis are, in this case, steady vector fields. This section is limited to discuss the topology of steady fields, due to the relevance to this thesis. Therefore, the term vector field topology is used in this thesis to denote the topology of steady vector fields, unless explicitly stated. Although, more recently, vector field topology has been extended to uncertain [Otto et al., 2011; Otto and Theisel, 2012] and time-dependent [Üffinger et al., 2013] vector fields. Other related work includes traditional topological visualization in constrained flow fields [Hauser and Gröller, 2000; Peikert and Sadlo, 2007].

Structural Stability

The vector field $\mathbf{u}$ is said to be *structurally stable* if for any other vector field $\mathbf{u}'$, such that $\mathbf{u}$ is sufficiently close to $\mathbf{u}'$, $\mathbf{u}'$ preserves the topology of $\mathbf{u}$. This analysis can be observed, for example, if we apply a small perturbation to $\mathbf{u}$. Such an analysis is fundamental to vector field topology, because steady vector fields are often approximations of real world vector fields with negligible, but not zero, variation over time. Therefore, features that may vanish with small perturbations are not subject of analysis in this context. Furthermore, *local structural stability* refers to the structural stability of subregions of the vector field, e.g., the vicinity of points or lines.

2.5.1 Critical Points

The first step to extract the topology of vector fields consist of finding and classifying their critical elements, i.e., *critical points* and *periodic orbits*. Critical points are isolated zeros in the field. Hence, given a critical point $\mathbf{x}_0$, $\mathbf{u}(\mathbf{x}_0) = \mathbf{0}$. The requirement for isolated zeros means that $\mathbf{u}$ is non-zero in the vicinity of $\mathbf{x}_0$.

Type	sorted eigenvalues
Source	$0 < \lambda_1 \leq \lambda_2 \leq \lambda 3$
Repelling saddle	$\lambda_1 < 0 < \lambda_2 \leq \lambda 3$
Attracting saddle	$\lambda_1 \leq \lambda_2 < 0 < \lambda 3$
Sink	$\lambda_1 \leq \lambda_2 \leq \lambda 3 < 0$

Table 2.1 — Possible types of first-order critical points according to the eigenvalues of the Jacobian, when they do not exhibit imaginary parts. The eigenvalues were sorted as follows $\lambda_1 \leq \lambda_2 \leq \lambda_3$. Figure 2.3 (top row) illustrates each one of these types.

Type	eigenvalues	
	Real	Complex
Spiral Source	$\lambda_1 > 0$	$\Re(\lambda_2) > 0$
Spiral Repelling Saddle	$\lambda_1 > 0$	$\Re(\lambda_2) < 0$
Spiral Attracting Saddle	$\lambda_1 < 0$	$\Re(\lambda_2) > 0$
Spiral Sink	$\lambda_1 < 0$	$\Re(\lambda_2) < 0$

Table 2.2 — Possible types of first-order critical points according to the eigenvalues of the Jacobian, when they consist of one real λ_1 and two complex conjugate λ_2 and λ_3. Note that λ_3 has been omitted in the table because $\Re(\lambda_2) = \Re(\lambda_3)$. Figure 2.3 (bottom row) illustrates each one of these types.

Critical points can then be classified according to the vector fields' behavior in its immediate vicinity, which can be identified using the first-order derivative at that point, i.e., the Jacobian matrix, here denoted

$$\mathbf{J}_{m,n} = \begin{bmatrix} \frac{\partial u_1}{\partial x_1} & \cdots & \frac{\partial u_1}{\partial x_m} \\ \vdots & \ddots & \vdots \\ \frac{\partial u_n}{\partial x_1} & \cdots & \frac{\partial u_n}{\partial x_m} \end{bmatrix}, \tag{2.6}$$

where m and n are here in accordance with Equation 2.1. If $\det(\mathbf{J}(\mathbf{x}_0)) \neq 0$, then $\mathbf{x}_0$ is a first-order critical point, and can be classified according to the real $\Re(\lambda_i)$

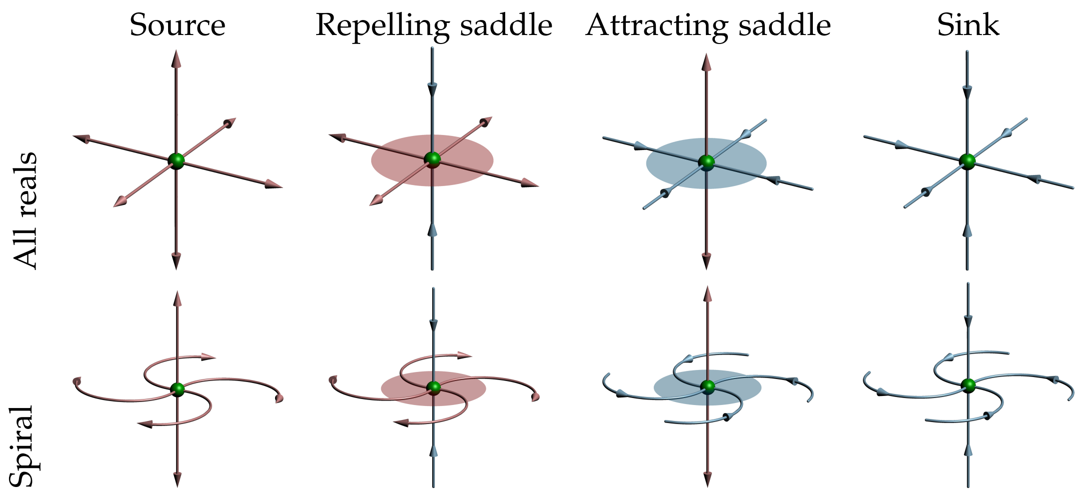


Figure 2.3 — Illustration of the types of first-order critical points. Top row shows the cases when the eigenvalues of the Jacobian are all real, and bottom row shows the types when two of the eigenvalues exhibit imaginary parts, called spirals. All saddle-type critical points (2-nd and 3-rd columns), combine attracting with repelling behavior of the field, and are the core of traditional vector field topology because they produce a seeding structure for separatrices of streamlines, i.e., 1D manifolds (2-nd column blue and 3-rd column red) and 2D manifolds (2-nd column red and 3-rd column blue).

and imaginary $\Im(\lambda_i)$ parts of the eigenvalues λ_i of $\mathbf{J}(\mathbf{x}_0)$, which provides insights about the field's behavior in its most representative directions around the critical point, i.e., the directions of the respective eigenvectors $\mathbf{e}_i$. For example, if λ_i exhibit imaginary part $\Im(\lambda_i) \neq 0$, then the vector field at an infinitesimally small offset from the critical point within the eigenplane spanned by the two complex eigenvectors exhibits swirling motion with respect to the critical point. According to their real parts, if $\Re(\lambda_i) > 0$ then the field is repelling in the direction of e_i, and it is attracting in this direction if $\Re(\lambda_i) < 0$. The field is neither repelling nor attracting if $\Re(\lambda_i) = 0$, however, hyperbolic critical points, i.e., those with non-zero real parts of eigenvalues, are of particular interest, because of their local structural stability.

In 3D vector fields, the three eigenvalues are either all real or one real and two complex conjugate. Thus, if they are all real, we can classify the critical point as *source*, *repelling saddle*, *attracting saddle*, and *sink*, according to Table 2.1. If they present imaginary parts, they can be classified as *spiral source*, *spiral repelling saddle*, *spiral attracting saddle*, and *spiral sink*, as shown in Table 2.2. Figure 2.3 illustrates the classes of critical points for all real (top row) and for one real and two complex conjugate eigenvalues (bottom row), accordingly.

Critical points can be considered a special, i.e., degenerate, kind of streamlines, and as such, are not intersected by those streamlines that actually converge to them in forward or reverse directions. Higher-order critical points, i.e., those where $\det(\mathbf{J}(\mathbf{x}_0)) = 0$, are out of the scope of this thesis, for such, the works by Scheuermann et al. [1997], Tricoche et al. [2000], and Weinkauf et al. [2005] are recommended as further reading.

In this work, the critical points are obtained according to Weinkauf [2008] by repeated subdivision of those cells whose vertices exhibit opposite sign in all vector components. After subdivision, the centers of the cells that still satisfy the sign criterion at the final subdivision level represent the critical points. The procedure is carried out on the GPU in parallel. We follow the common practice to estimate the Jacobian at the grid nodes and to interpolate it at the critical points.

2.5.2 Periodic Orbits

A periodic orbit is an isolated solution of $\mathbf{x}(\tau)$ (Equation 2.4), such that there exist a pair of values (τ_i, τ_j) where $\tau_i \neq \tau_j$, $\mathbf{x}(\tau_i) = \mathbf{x}(\tau_j)$ and $\mathbf{x}(\tau)$ is not constant. If $\mathbf{x}(\tau)$ is constant, then it is a critical point. In other words, a periodic orbit is an isolated cyclic streamline, and it is here denoted $\mathbf{x}_c(\tau)$. Wischgoll and Scheuermann [2002] extract periodic orbits by detecting and analyzing streamlines that return to cells of the grid and a recent technique by Kasten et al. [2014] extracts saddle-type periodic orbits based on the Finite-Time Lyapunov Exponent (FTLE). Chapter 3 shows that saddle-type periodic orbits can also be extracted by means of bifurcation line extraction.

Unlike critical points, periodic orbits cannot be locally classified. Therefore, their classification is done due to the Jacobian of Poincaré maps, also called first-recurrence maps. Poincaré maps consist of lower-dimensional sections, called Poincaré sections, here denoted S, that intersect the periodic orbit anywhere and that are transversal to the vector field, i.e., there must not be any point in S such that S is tangent to $\mathbf{u}$. Then, the Poincaré map is given by $\mathbf{p} : S_0 \to S$ with $S_0 \subset S$, that maps points $\mathbf{x} \in S_0$ to points $\mathbf{p}(\mathbf{x}) \in S$, such that $\mathbf{p}(\mathbf{x})$ is the first intersection of S and the streamline $\mathbf{x}(\tau)$ with the initial condition $\mathbf{x}(\tau_0) = \mathbf{x}$, which consists of a full revolution of $\mathbf{x}(\tau)$ after $\mathbf{x}$ with respect to $\mathbf{x}_c$ (see Figure 2.4).

An interesting observation is that the topology of Poincaré maps is consistent with the topology of streamlines in the vicinity of periodic orbits. Thus, periodic orbits can be classified according to the eigenvalues of the Jacobian of Poincaré maps $\mathbf{J}(\mathbf{p})$. In 3D vector fields, the Poincaré section S is a bidimensional space, and assuming that $\mathbf{x}$ and $\mathbf{p}(\mathbf{x})$ are given in a 2D coordinate system in S, for

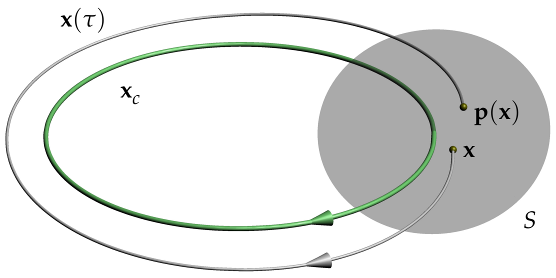


Figure 2.4 — Illustration of Poincaré maps. The periodic orbit (green curve) intersects the Poincaré section S (gray disc), that is nowhere tangent to the vector field. The Poincaré map at $\mathbf{x}$ is computed by integrating a full revolution of the streamline (white curve) seeded at $\mathbf{x}$, which gives the point $\mathbf{p}(\mathbf{x})$. Poincaré maps are used to classify periodic orbits, because their topology is consistent.

example, with the origin at the intersection between $\mathbf{x}_c$ and S, then $\mathbf{J}(\mathbf{p})$ has two eigenvalues λ_i, which can be either both real or complex conjugate. Hyperbolic periodic orbits (i.e., those that are locally structurally stable) are then identified when the eigenvalues are both off the unit circle of the complex plane ($|\lambda_1| \neq 1$ and $|\lambda_2| \neq 1$). If both eigenvalues are real, the periodic orbits are classified as *source*, *sink*, *saddle*, and *twisted saddle* periodic orbits, according to Table 2.3, and if they are two complex conjugates, they are classified as *spiral source* and *spiral sink* periodic orbits according to Table 2.4. Figure 2.5 shows illustrations of the types of periodic orbits. Analogous to the critical points, the periodic orbits of types saddle and twisted saddle (center column) are singularities where the flow splits and, hence, provide two separating 2D manifolds of streamlines that converge to them in positive and negative time, respectively.

Type	eigenvalues	
Source	$\|\lambda_1\| \geq \|\lambda_2\| > 1$	
Saddle	$\|\lambda_1\| > 1 > \|\lambda_2\|$	and $\lambda_1, \lambda_2 > 0$
Twisted saddle	$\|\lambda_1\| > 1 > \|\lambda_2\|$	and $\lambda_1, \lambda_2 < 0$
Sink	$1 > \|\lambda_1\| \geq \|\lambda_2\|$	

Table 2.3 — Possible types of periodic orbits due to the eigenvalues of the Jacobian of the Poincaré map, when they are both real. To this table the eigenvalues were sorted as follows $\lambda_1 \leq \lambda_2$. According to Asimov [1993], the product between λ_1 and λ_2 is always positive, thus, eigenvalues with opposite signs are not possible. For illustrations of the types, see Figure 2.5.

Type	eigenvalues
Spiral source	$\|\lambda_1\| > 1$
Spiral sink	$\|\lambda_1\| < 1$

Table 2.4 — Possible types of periodic orbits according to the eigenvalues of the Jacobian of the Poincaré map, when they are both complex conjugate λ_1 and λ_2. For illustrations of the types, see Figure 2.5.

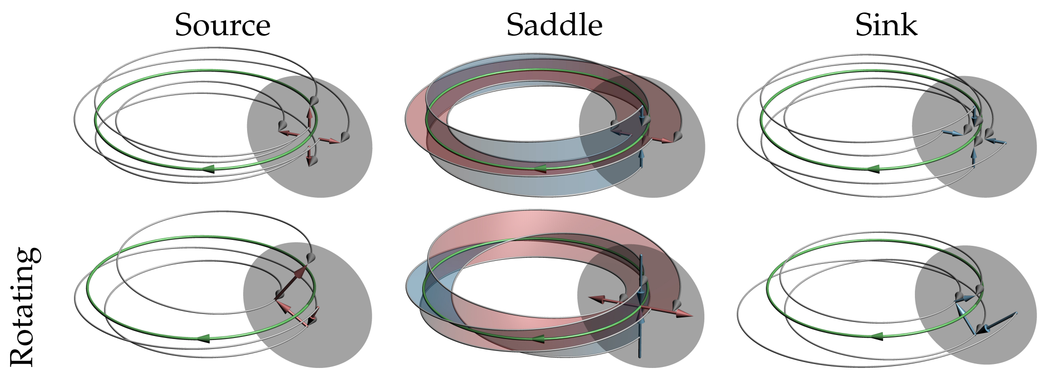


Figure 2.5 — Illustration of the types of periodic orbits. Top row shows from left to right source, saddle, and sink periodic orbits, while bottom row shows from left to right spiral source, twisted saddle, and spiral sink periodic orbits. One can see the relation between the streamlines (white curves) in the vicinity of the periodic orbits (green) with the Poincaré maps (red and blue arrows in gray disks).

2.5.3 Manifolds

A stable manifold of a hyperbolic critical element $\mathbf{x}_h$ (i.e., critical points or periodic orbits), denoted $W^s(\mathbf{x}_h)$, consist of all points from which the streamlines $\mathbf{x}(\tau)$ converge to $\mathbf{x}_h$ with $\tau \to \infty$, while the unstable manifold, denoted $W^u(\mathbf{x}_h)$, are the points from which $\mathbf{x}(\tau)$ converge to $\mathbf{x}_h$ with $\tau \to -\infty$. Therefore, given the classification of critical points in 3D vector fields (Figure 2.3), we identify manifolds with different dimensionalities. For example, the critical points of types sink and source have 3D stable and 3D unstable manifolds, respectively, because they exhibit streamlines converging to them in positive and negative time, respectively, from all directions. The critical points of type saddle, on the other hand, exhibit either one direction converging to them and one plane of directions diverging from them, defined by two axes of divergence, or one direction diverging from them and one plane of directions converging to them, defined by two axes of convergence. Hence, saddle-type critical points

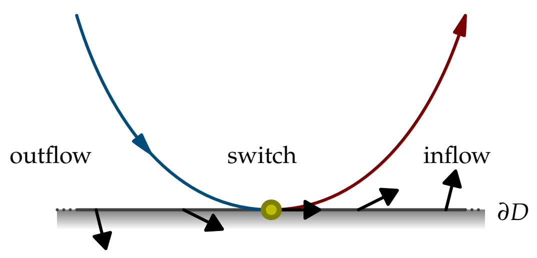


Figure 2.6 — Illustration of an inbound boundary switch (yellow), which represents the point at the boundary ∂D where the vector field is tangent to ∂D and pointing to an inflow region. Black arrows at the boundary illustrate the vector field. The manifolds (blue/red) from the inbound boundary switch point separate the vector field into outflow, switch, and inflow regions, with respect to the boundary.

define either one 1D stable manifold and one 2D unstable manifold or one 1D unstable manifold and one 2D stable manifold. Analogous reasoning applies to manifolds of periodic orbits. Thus, periodic orbits of type sink and source define 3D stable and 3D unstable manifolds, while periodic orbits of type saddle define two 2D manifolds, one stable and one unstable. Note that, following the aforementioned definition, the points at the critical elements are part of their manifolds, i.e., $\mathbf{x}_h(\tau) \in W^s(\mathbf{x}_h)$ and $\mathbf{x}_h(\tau) \in W^u(\mathbf{x}_h)$.

In visualization of 3D vector field topology, we often aim the visualization of the global structure of the vector fields with respect to their 3D manifolds, which, because of occlusion, is usually achieved by extracting the separation between those regions (i.e., separatrices), that are the 1D and 2D manifolds. However, also due to occlusion, even the 2D manifolds provide in many situations very complex visualizations. Nevertheless, as stated by Asimov [1993], 2D stable manifolds from one critical element can intersect 2D unstable manifolds from either an other critical element or from itself. Since 1D and 2D manifolds consist of streamlines, their intersection again is a streamline, which connects those critical elements. Therefore, Theisel et al. [2003] proposed the visualization of the topology based on these intersections, called *saddle connectors*, which has the advantage to reduce occlusion and preserve the perception of the topology. A further conceptual contribution to this field are the connectrices [Bachthaler et al., 2012], that instead of showing field separation, they simplify the visualization by extracting the flux connectivity between dipoles in 2D magnetic fields.

2.5.4 Boundary Switches

In bounded domains or when one aims the analysis of the transport in subregions, the critical points or periodic orbits can be at least partially located outside the subregion, which sometimes makes it impossible to extract the topology based on the traditional approach. Therefore, Scheuermann et al. [2003] proposed an approach in 2D that detects switches of outflow to inflow in the boundaries of the domain or subregion, called *inbound boundary switches*, and compute the manifolds that converge to them. Weinkauf et al. [2004] extended this approach, afterwards, for 3D domains.

Given the domain D, boundary switches are, hence, detected as the loci at the domain boundaries ∂D where the vector field is tangent to the boundaries. They are further classified as inbound and outbound, depending on whether the vector field at the boundary switches is pointing from an outflow to an inflow region of ∂D or from an inflow to an outflow region, respectively. Inbound boundary switches are of particular interest, because they define streamlines as separatrices of the vector field with respect to the domain boundaries, as they separate regions of the vector fields that flow outwards the domain in positive or negative time, from those that do not reach that part of the domain boundary. The outbound boundary switches represent the same behavior but from outside the domain boundaries, and are, hence, of no relevance with this respect. Boundary switch curves are simply the counter part to boundary switches in 3D vector fields. Although they provide relevant insights to the topology of vector fields in bounded domains, boundary switches do not represent splits in the essential meaning, and their results depend on the boundary, e.g., changing the position of the boundary also changes the boundary switches. Bifurcation lines, on the other hand, to which an extraction technique is presented in Chapter 3, provide vector field separation in the absence of critical points and periodic orbits but are independent of boundary configuration. Figure 2.6 illustrates an inbound boundary switch (yellow) and how the manifolds from it separate the vector field into outflow, switch, and inflow regions of the vector field. In analogy to the saddle connectors [Theisel et al., 2003], Weinkauf et al. [2004] extracted the so-called *boundary switch connectors*, which consist of the streamlines at intersections between manifolds from boundary switches and manifolds from either other boundary switches or any critical element.

2.6 Flow Features

Feature extraction represents a subfield of vector field visualization, that concentrates on detecting specific structures of interest in vector fields for their direct investigation. This section discusses some approaches that are related to the projects presented in this thesis. For an overview of feature extraction from vector fields, Post et al. [2003] is recommended as further reading.

2.6.1 Parallel Vectors Operator

Many line-type features can be formulated as the set of loci where two (derived) vector fields are parallel (note that throughout this thesis we use the term parallel, denoted as $\parallel$, also for the antiparallel configuration). In their framework, called the parallel vectors operator, Peikert and Roth [1999] formulate different line-type features by means of this approach and provide an algorithm for their efficient extraction. The framework consists of two vector fields $\mathbf{v}$ and $\mathbf{w}$ and solves for the set of points where $\mathbf{v} \parallel \mathbf{w}$, formulated as

$$\mathbf{v} \times \mathbf{w} = \mathbf{0}\,. \tag{2.7}$$

In this thesis, the "analytic solution for triangulated faces" approach from [Peikert and Roth, 1999] was implemented, i.e., non-triangular cell faces are triangulated and the solutions of Equation 2.7 are solved for thereon, as this approach is faster and typically provides the same results as their approach based on Newton-Raphson iteration.

Once these "raw" features are obtained, one typically needs to suppress false positives (noise). In this thesis, the approach by Peikert and Roth [1999] is followed, and those parts of features where the angle $\alpha(\boldsymbol{\vartheta}, \mathbf{v}) := \cos^{-1}(|\boldsymbol{\vartheta} \cdot \mathbf{v}|/(\|\boldsymbol{\vartheta}\| \cdot \|\mathbf{v}\|))$, defined as the angle between the vector field $\mathbf{v}$ and the tangent $\boldsymbol{\vartheta}$ of the feature line, exceeds the user-defined threshold $\tau_\alpha \geq 0$ are rejected. The rationale behind this is that many line-type features conceptually represent streamlines of $\mathbf{v}$ or $\mathbf{w}$, like the bifurcation lines in Chapter 3. Note that in typical datasets τ_α has to be chosen comparably large, because the criteria often have problems in detecting the exact location of the feature. As a second filtering approach, which is applied subsequently, a minimum length τ_μ of the features according to Peikert and Sadlo [2008] is required, to suppress short lines that represent erroneous or weak features. The result of the procedure is a set of polylines.

2.6.2 Vortices

In the literature, there is no consensus on a formal definition for *vortices* that holds in all configurations, and a rough definition for vortices is, though, the regions where the vector field exhibits swirl behavior. A challenging problem on the extraction of vortices is to detect their axis of rotation, called *vortex core*

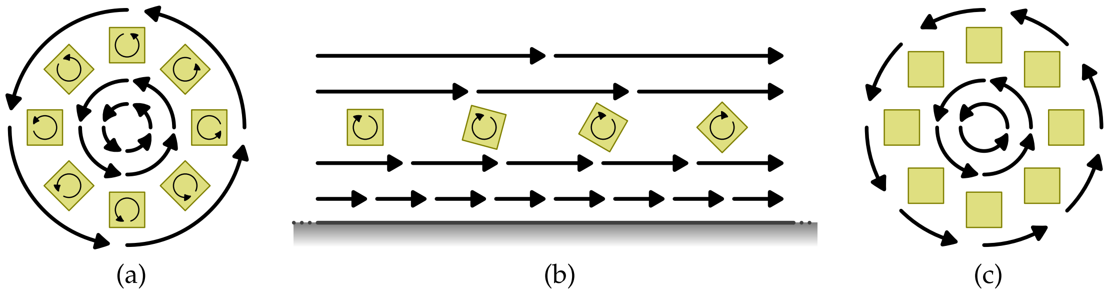


Figure 2.7 — Illustrations of vorticity $\boldsymbol{\omega}$ on (a) rigid-body-like vortex, (b) shear flow, and (c) irrotational vortex. The length of arrows represents vector magnitude, which can be interpreted, for example, as force or velocity, and the yellow squares illustrate local rotation of the flow. In rigid-body-like vortices (a) the vorticity is constant, which is illustrated by the squares rotating around their own axis at the same angular velocity as they are rotating around the flow's center of rotation. Shear flow (b) does not exhibit actual rotation of the flow, but still exhibits non-zero vorticity. In irrotational vortices (c) rotation is compensated by shear and they exhibit zero vorticity, except for the center of rotation that has high vorticity.

lines, as they provide a concise representation of the vortical flow. Vortex core lines are closely related to vector field topology, but usually not considered topological constructs.

Various approaches have been presented so far for the detection of vortices and extraction of vortex core lines. Villasenor and Vincent [1992] detect vortices as those regions with high *vorticity* $\boldsymbol{\omega}$, which is given by the curl of the vector field as $\boldsymbol{\omega} = \nabla \times \mathbf{u}$ and indicates local rotation. Note however that, actual rotations are not directly related to non-zero vorticity. For example, regions with non-zero vorticity can also be found in straight parallel flows like shear flow, which often happens in fluid flows close to no-slip boundaries (Figure 2.7b). Moreover, regions with zero vorticity may be found in regions with actual rotation, for example, the irrotational vortices (Figure 2.7c). Furthermore, the vorticity in rigid-body-like vortices (Figure 2.7a) is constant. Sujudi and Haimes [1995] extract vortex core lines as those loci where the velocity is parallel to the real eigenvector of the velocity gradient with the additional requirement that the other two eigenvectors are complex. His approach was extended for time-dependent vector fields by Weinkauf et al. [2007] and Fuchs et al. [2008]. Roth and Peikert [1996] identified vortex core lines as those locations where vorticity is parallel to velocity. Levy et al. [1990] extract vortices based on *helicity*, which is the dot product between velocity and vorticity $\mathbf{u} \cdot \boldsymbol{\omega}$. According to their work, considering *normalized helicity*, i.e., $\|\mathbf{u}\| \cdot \|\boldsymbol{\omega}\|$, improves the result over non-normalized by rejecting shear flow (see Figure 2.7b). Roth and Peikert [1998] extended the approach by Sujudi and Haimes [1995] to higher order. They identified that the previous approach [Sujudi and Haimes, 1995] represents the locations where streamlines are straight, reformulated it as the points where acceleration is parallel to velocity, and based on this they extended the approach to a higher order by formulating a criterion that extracts those points of streamlines that exhibit zero torsion. Chapter 3 modifies both approaches, that by Sujudi and Haimes [1995] and that by Roth and Peikert [1998] for the extraction of bifurcation lines. Sahner et al. [2005] define vortex core lines as valley lines of λ_2, a scalar field introduced by Jeong and Hussain [1995] indicating vortex regions, where negative. A similar approach was presented by Schafhitzel et al. [2008], which constrains valley lines of λ_2 with regions consistent with λ_2 isosurfaces.

2.6.3 Attachment and Separation Lines

In vector fields, *attachment lines* are curves at physical boundaries that have manifolds of off-boundary streamlines converging to them. *Separation lines* are likewise defined but in negative time direction. Thus, attachment and separation lines have also stable and unstable manifolds, respectively. These lines are of particular interest in the study of aerodynamics, because they are related to flow separation, e.g., leading to stall, and, therefore, should be prevented [Post et al., 2003; Sadlo, 2010]. Helman and Hesselink [1991] use these manifolds to complement the traditional vector field topology with respect to boundaries. Kenwright et al. [1999] proposed two algorithms for automatic extraction of attachment and separation lines, of both types open and closed, different from previous approaches in that they achieve a local extraction. The attachment and separation lines of type closed are characterized by starting and ending on critical points, while the lines of type open are not necessarily related to critical points. Detection of open lines is, therefore, more difficult.

CHAPTER 3

Bifurcation Lines

Bifurcation lines, as proposed by Perry and Chong [1987], represent streamlines that exhibit, locally for the longest time, one converging and one diverging 2D manifold of streamlines and, surprisingly, have not yet been sufficiently recognized in scientific visualization. Figure 3.1 illustrates a bifurcation line (green) with the respective stable and unstable manifolds (blue and red, respectively). In his thesis, Roth [2000] states that bifurcation lines could be obtained by extending the local criterion proposed by Kenwright et al. [1999] to 3D vector fields, which was originally defined for attachment and separation lines on surfaces (Section 2.6.3). Roth also mentions that the resulting 3D criterion is identical to the vortex core line criterion by Sujudi and Haimes [1995] (Section 2.6.2), with the only difference that instead of requiring complex eigenvalues of the velocity gradient, real eigenvalues are required. Finally, he also notes that this extraction technique would work only for sufficiently straight bifurcation lines, for the same reasons that motivated his higher-order approach for the extraction of vortex core lines [Roth and Peikert, 1998]. As stated in that paper, however, this higher-order extraction technique can provide better results in some cases but may fail to provide a solution in other cases.

In this chapter, both vortex core line extraction techniques, that of Sujudi and Haimes [1995] and that by Roth and Peikert [1998], are modified and examined for the purposes of extracting bifurcation lines. Beyond that, the parallel vectors operator due to Peikert and Roth [1999] is used and a technique for the refinement of the resulting curves toward the aimed bifurcation lines is presented. Beyond that, an approach to construct a seeding structure that

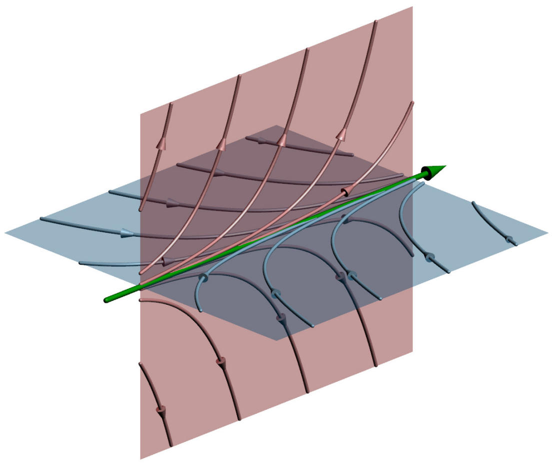


Figure 3.1 — Illustration of a bifurcation line (green, according to [Perry and Chong, 1987]), with two 2D manifolds of streamlines, one converging to the bifurcation line in positive (blue), and one in negative (red) time.

extracts the 2D manifolds of bifurcation lines is presented. One of the results is that in the experiments, as expected, the modified criterion by Roth and Peikert [1998] provided better results (i.e., required less refinement) than the modified criterion by Sujudi and Haimes [1995], but there were many cases where the modified criterion by Roth and Peikert [1998] failed to generate a result at all. Although in many of these cases the modified criterion by Sujudi and Haimes [1995] still generated inaccurate results, these solutions after the refinement approach, here presented, converge to the bifurcation line and provide accurate results.

Bifurcation lines are closely related to the concept of saddle connectors (Section 2.5.3), which are the intersections of two distinct 2D manifolds from saddle-type critical points. Saddle connectors represent bifurcation lines when close to the respective critical points (Figures 3.2a and 3.2b), since there they are part of the hyperbolic dynamics of the saddle point. In other words, the 1D manifold of saddle-type critical points is always part of the respective 2D manifold of a bifurcation line (Figure 3.2b). Furthermore, because saddle connectors are streamlines, the manifolds from bifurcation lines must in continuous representations reconstruct the manifolds of saddle connectors and consequently those of the critical points. For instance, in Figure 3.2 the essential difference between the manifolds from critical points in (a) and the manifolds from bifurcation lines in (b) are the gaps emanating from critical points. Those gaps are direct consequence of discretization, and they would degenerate to gaps as wide as a single streamline in the continuous case.

A final observation is that saddle-type periodic orbits (Section 2.5.2) resemble closed saddle connectors, however, without a critical point along them. For this reason, saddle periodic orbits cannot be obtained by intersecting 2D manifolds

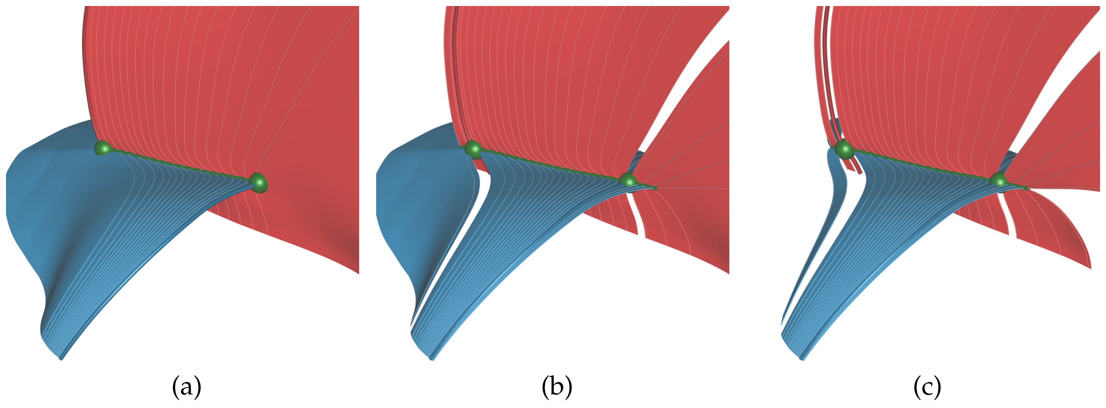


Figure 3.2 — Solar MHD dataset. (a) Saddle connector (green line) represents intersection curve between stable 2D manifold (blue) of left critical point (green point) and unstable 2D manifold (red) of right critical point (green point). 2D manifolds are limited by the respective 1D manifolds (dark red and dark blue lines) of the other saddle-type critical point. (b) Same view but with manifolds of bifurcation lines (green lines). 1D manifolds of critical points are consistent with 2D manifolds of bifurcation lines. Due to discretization, the bifurcation line and thus also its 2D manifolds exhibits gaps at the critical points which can be avoided by connecting bifurcation lines across critical points. (c) Same as (b) but without seed curve connection (Fig. 3.8) at inflow and outflow side of bifurcation lines. This would lead to unnecessary gaps in the manifolds.

of saddle-type critical points. Nevertheless, in analogy to saddle connectors, at least a part of a saddle-type periodic orbit has to represent a bifurcation line, too (Figure 3.4). While one might think that obtaining only parts of saddle connectors and saddle-type periodic orbits represents a limitation, rather the opposite is the case. Bifurcation lines provide insight into the hyperbolic dynamics of a vector field with respect to its topological structure, similar to saddles. Beyond that, bifurcation lines can be present in absence of critical points, saddle connectors, or saddle-type periodic orbits, complementing traditional vector field topology as one can see in Figures 3.2, 3.3, 3.11, and 3.13.

All in all, in vector fields defined on unbounded domains, parts of bifurcation lines could be determined by marking each point of saddle connectors and periodic orbits that exhibits hyperbolic behavior. Although this approach would require the costly extraction of periodic orbits and would involve comparably expensive integration of streamlines, it could work for such vector fields. However, it has to be noted that accurate numerical integration of streamlines in hyperbolic regions, i.e., along bifurcation lines (or hyperbolic trajectories [Haller,

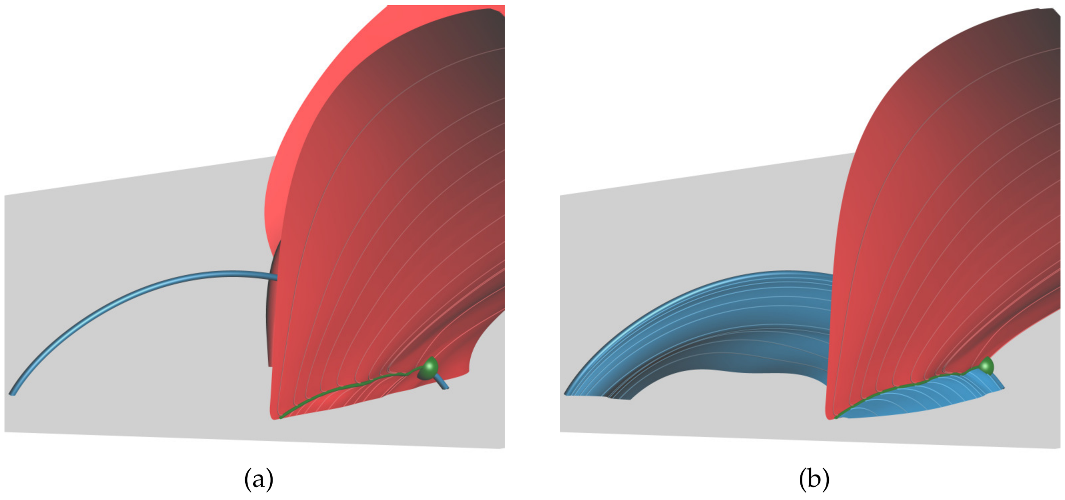


Figure 3.3 — Solar MHD dataset. (a) Visualization with traditional vector field topology. Saddle-type critical point (green point) is close to domain boundary (gray). The second saddle-type critical point that would be needed for obtaining the saddle connector is located outside the domain, and therefore not available. Hence, the 1D manifold (blue) and the 2D manifold (red) of only one saddle point provide only limited insight on the topology. (b) 2D manifolds (red and blue) of the bifurcation line (green line in (a) and (b), touching the boundary at the bottom of the image) provide full topological insight.

2000]), is often a very difficult undertaking because in either, forward or reverse, direction of integration the tangent curve is repelled from the bifurcation line as integration errors are growing exponentially. The white streamline in Figure 3.4a exemplifies this kind of problem with a saddle-type periodic orbit that exhibits medium hyperbolicity along it. There, finding a seed point that approximates the streamline (white) to the periodic orbit (green) was not successful. Note that, periodic orbits with high hyperbolicity also impairs the applicability of Poincaré maps (Section 2.5.2). In contrast, a technique that performs local extraction of bifurcation lines, as presented in this chapter, captures the entire orbit. It has to be noted, however, that saddle orbits with very low hyperbolicity can be integrated and that periodic orbits are also amenable to identification with Poincaré maps, for example, in Figure 3.4, the Poincaré map (gray) illustrated in (d) represents the identification of the periodic orbit in (b).

The situation is, however, different in vector fields on bounded domains or if the independent analysis of subregions is of interest. There, due to the traditional vector field topology, one or both saddle-type critical points that are necessary for the extraction of saddle connectors, or also part of periodic

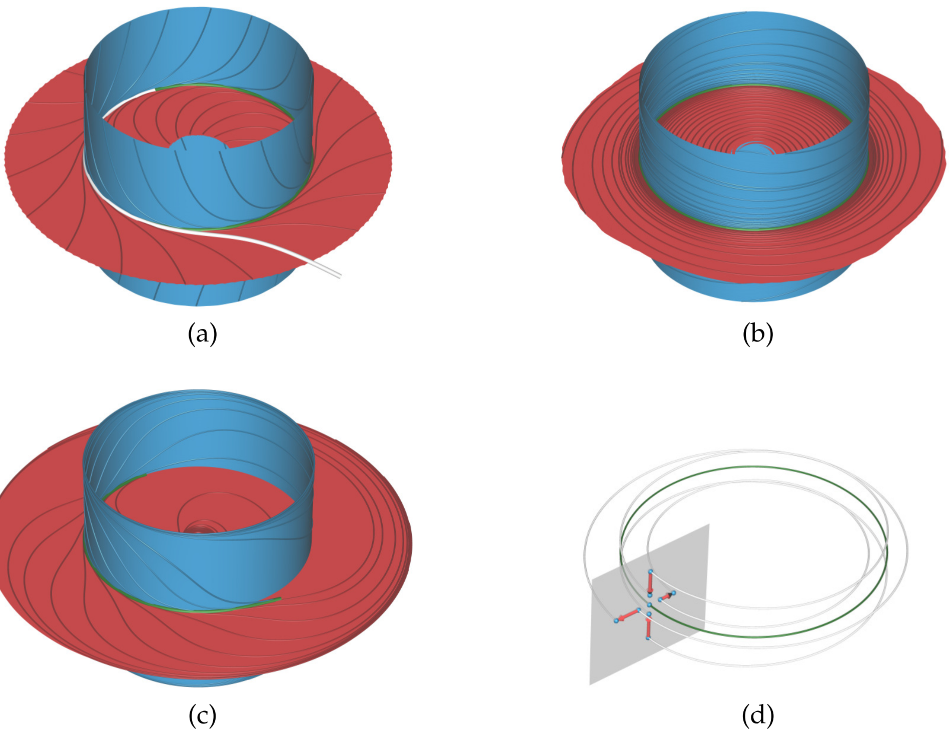


Figure 3.4 — Saddle Orbit datasets. (a) Slow orbit (I) (rotation 0.4) (cf. Figure 3.5a). Periodic orbit is captured as a bifurcation line with refined $\mathbf{u} \parallel \mathbf{a}$ (green), providing stable (blue) and unstable (red) manifolds. Due to the substantial hyperbolicity, obtaining the periodic orbit by integration of a streamline (white curve) was not successful. (b) Fast periodic orbit (IV) (rotation 10.0). Periodic orbit is captured as bifurcation line with refined $\mathbf{u} \parallel \mathbf{b}$ (green). (c) Periodic orbit (VI) that exhibits high hyperbolicity at the left side but zero hyperbolicity at the right side. Only the sufficiently hyperbolic (left) part of the periodic orbit is extracted as bifurcation line, which is obtained with refined $\mathbf{u} \parallel \mathbf{a}$ ($\mathbf{u} \parallel \mathbf{b}$ failed). Note, however, that the right end of the bifurcation line deviates, i.e., the refinement aligned it with a streamline different to the bifurcation line. Nevertheless, this has negligible impact, since the 2D manifolds are obtained using "connected" seeding curves (see Figure 3.8), resulting in complete 2D manifolds. (d) Analysis of the periodic orbit (b) with a Poincaré map (gray).

orbits, may be located outside the domain. In all these cases, it would be impossible to obtain the respective topological constructs based on streamline integration, as one can see in Figure 3.3a. The boundary switch connectors (Section 2.5.4), presented by Weinkauf et al. [2004], are an alternative to saddle connectors in bounded domains. The results of constructs, however, depend on the domain boundary, while a technique that performs local extraction of bifurcation lines, in contrast, does not make a difference if part of a bifurcation line, saddle connector, or periodic orbit is located outside the domain. Figure 3.3 exemplifies this kind of problem. There, the bifurcation line (green) provides full topological insight (b), while the traditional vector field topology (a) is limited by the manifolds computed from only one critical point, and provides only partial topology.

Furthermore, according to Perry and Chong [1987], bifurcation lines can also be unrelated to any further features, in other words, they can start and end without relation to other singularities. Bifurcation lines provide local insights on the topology of vector fields, due to their hyperbolic behavior, but precaution shall be used when performing such analysis based on their manifolds. In fact, as long as a bifurcation line is related to critical points or periodic orbits, they are likely to be consistent with the global topology of the vector field. Moreover, once the bifurcation lines reach the boundaries, they are further candidates for the analysis of bounded vector field topology. Because bifurcation lines represent streamlines, all bifurcation lines are likely to represent at least one of these cases. However, the scope of this work is on their extraction, and further investigation is needed on their application.

In this chapter, modified versions of the vortex core lines extraction approaches of Sujudi and Haimes [1995] and Roth and Peikert [1998] are used for local extraction of bifurcation lines. In this respect, the union of the respective vortex core lines, bifurcation lines, and constructs from traditional vector field topology represents an extended topological representation of vector fields.

3.1 Extraction of Bifurcation Lines

In this section, the technique for local extraction of bifurcation lines is presented, which consists of two basic steps: first, an initial solution by the extraction with the parallel vectors operator using modified vortex core line criteria (Section 3.1.1) and, then, the refinement of this solution toward a streamline (Section 3.1.2).

3.1.1 Modification of Vortex Core Line Criteria

As described in Section 2.6.2, the criterion presented by Sujudi and Haimes [1995] identifies a vortex core line in 3D vector fields where velocity $\mathbf{u}$ is parallel to the real eigenvector of the velocity gradient (Equation 2.6) $\nabla\mathbf{u}$, or in other words, $\mathbf{u}$ is parallel to the steady formulation of acceleration $\mathbf{a} := (\nabla\mathbf{u})\mathbf{u}$,

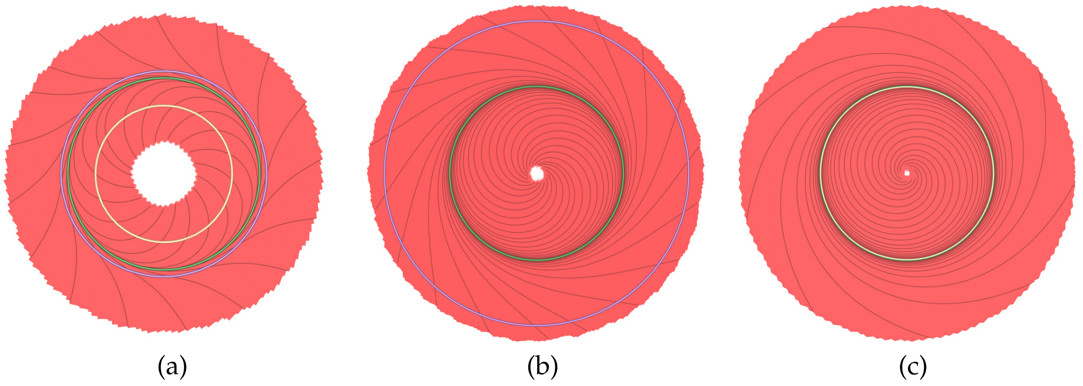


Figure 3.5 — Saddle Orbit datasets. View on the planar 2D unstable manifold (red). The solutions of $\mathbf{u} \parallel \mathbf{a}$ and $\mathbf{u} \parallel \mathbf{b}$ are colored purple and yellow, respectively, and the solutions of $\mathbf{u} \parallel \mathbf{a}$ or $\mathbf{u} \parallel \mathbf{b}$ after applying the refinement approach are colored green. (a) Slow orbit (I) (rotation 0.4) (cf. Figure 3.4a). Solution of $\mathbf{u} \parallel \mathbf{a}$ is close to refined $\mathbf{u} \parallel \mathbf{a}$, while $\mathbf{u} \parallel \mathbf{b}$ deviates substantially from the periodic orbit due to strong hyperbolicity. (b) Faster orbit (II) (rotation 1.3) (cf. Figure 3.7b). Solution of $\mathbf{u} \parallel \mathbf{a}$ deviates substantially but $\mathbf{u} \parallel \mathbf{b}$ provides no solution at all, while refined $\mathbf{u} \parallel \mathbf{a}$ obtains correct solution (cf. Figure 3.7b). (c) Fast orbit (III) (rotation 3). $\mathbf{u} \parallel \mathbf{a}$ fails while $\mathbf{u} \parallel \mathbf{b}$ without refinement is very close to $\mathbf{u} \parallel \mathbf{b}$ with refinement.

resulting in

$$\mathbf{u} \parallel \mathbf{a}\,, \tag{3.1}$$

with the additional requirement that the other two eigenvectors (and hence also the other two eigenvalues) exhibit non-zero imaginary parts.

Roth states in his thesis [Roth, 2000] that if one instead requires all eigenvalues to be real, the resulting curves would include bifurcation lines, assuming that the bifurcation lines are sufficiently straight just as for the vortex core lines. Since Equation 3.1 in this case holds if *any* of the three eigenvectors of $\nabla\mathbf{u}$ is parallel to $\mathbf{u}$, one needs to test for bifurcation line behavior. As illustrated in Figure 3.1, a bifurcation line is characterized by hyperbolic behavior in sections perpendicular to the line, i.e., there must be one direction that converges toward the line in forward and one in reverse time if the vector field is projected to that plane. Such a hyperbolic behavior can be characterized in 2D flow by $\det(\nabla\mathbf{u}) < 0$. Hence, besides rejecting feature points where the angle $\alpha(\boldsymbol{\vartheta}, \mathbf{u})$ between the tangent $\boldsymbol{\vartheta}$ of the extracted line and the vector field exceeds a user-defined threshold τ_α (see Section 2.6.1), a minimum feature strength $\tau_\chi > 0$ is additionally required, i.e., one projects $\nabla\mathbf{u}$ onto the plane perpendicular to $\mathbf{u}$, computes its determinant d and rejects the feature point if $d/\|\mathbf{u}\| > -\tau_\chi$. Finally, those lines shorter than a user-defined minimum length τ_μ (see Section 2.6.1) are rejected. Please note that this approach is here denoted as $\mathbf{u} \parallel \mathbf{a}$ or simply Equation 3.1. Example results are shown by the purple curves in Figure 3.5 in the context of the extraction of saddle-type periodic orbits using bifurcation lines.

The higher-order criterion for vortex core lines due to Roth and Peikert [1998] identifies vortex core lines as those loci where velocity $\mathbf{u}$ is parallel to the steady formulation of the jerk vector $\mathbf{b} := (\nabla\mathbf{a})\mathbf{u} = (\nabla((\nabla\mathbf{u})\mathbf{u}))\mathbf{u}$, resulting in

$$\mathbf{u} \parallel \mathbf{b}\,, \tag{3.2}$$

with the additional requirement that $\nabla\mathbf{u}$ exhibits one real and two complex eigenvalues. While Equation 3.1 originally identifies points with zero streamline curvature as belonging to a vortex core line, Equation 3.2 extends the requirement to zero streamline torsion, i.e., locally planar streamlines. The motivation for this definition is that the solutions of Equation 3.1 are displaced from the aimed vortex core line the more this core line is curved and the slower the flow rotates around the core line compared to the velocity along the core line. In other words, Equation 3.1 fails for bent vortices, in particular if they exhibit a high longitudinal speed. Interestingly, this is also the case for the derived bifurcation line criterion with Equation 3.1, as demonstrated in Figure 3.5. The reason for this is that it identifies the points of zero streamline curvature (see also blue curve in Figure 3.6a). It is apparent in Figure 3.5, that Equation 3.2 performs substantially better, however, at the drawback that it may fail at providing a solution.

Here, the approach of Roth and Peikert [1998] is accordingly modified for bifurcation lines, i.e., as in the case of the criterion by Sujudi and Haimes [1995], it is required that all eigenvalues are real. Besides the difference in the parallel vectors definition, i.e., Equation 3.2 instead of Equation 3.1, the procedure is identical. This modified technique is denoted as $\mathbf{u} \parallel \mathbf{b}$ or simply Equation 3.2. Example results are shown by the yellow curves in Figure 3.5 on the extraction of saddle-type periodic orbits as bifurcation lines.

3.1.2 Refinement of Parallel Vectors Solutions

As demonstrated in Figure 3.5, $\mathbf{u} \parallel \mathbf{b}$ is often more accurate than $\mathbf{u} \parallel \mathbf{a}$. In fact, it was observed that it provides very accurate results for high velocities (relative to hyperbolicity) along the bifurcation line. However, there are many cases where it fails to provide a result at all (e.g., Figures 3.5b, 3.9, 3.10, 3.11, and 3.13). Some of these cases are amenable to extraction by $\mathbf{u} \parallel \mathbf{a}$, however

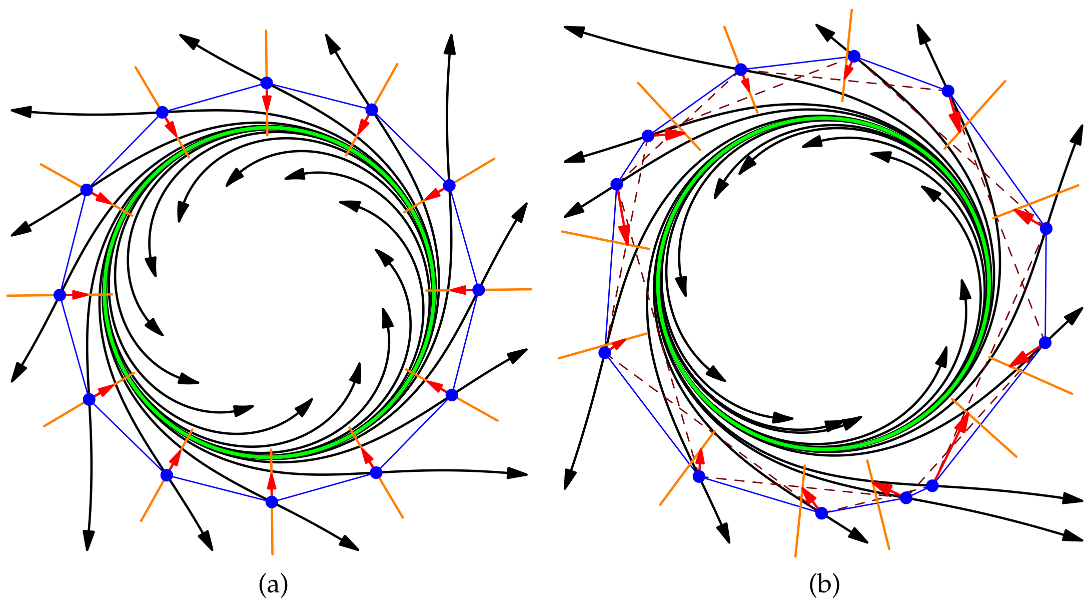
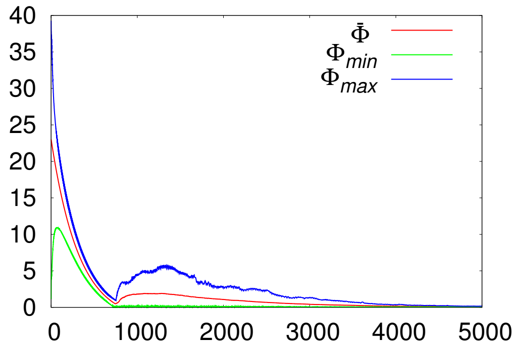


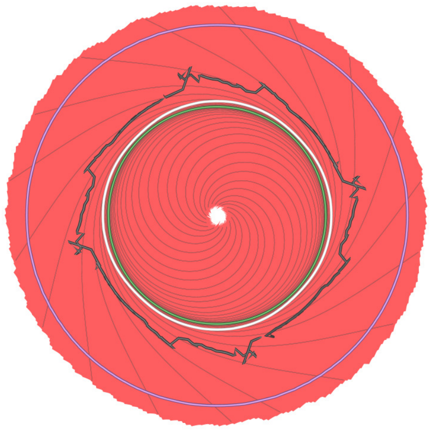
Figure 3.6 — (a) Illustration of the algorithm for bifurcation line refinement at the example of the radial manifold from Figure 3.4a: the bifurcation line (here represents a saddle-type periodic orbit) (green), solution of zero streamline curvature $\mathbf{u} \parallel \mathbf{a}$ (blue), planes normal to tangent of blue curve (orange), and correction vector (red). Small steps along the correction vectors are applied iteratively, making the blue curve converge toward the bifurcation line. (b) Same as (a) illustrating impact of the regularization in case of uneven distribution of vertices along the blue polyline.

with the drawback that $\mathbf{u} \parallel \mathbf{a}$ does not provide sufficient accuracy for practical applications such as the extraction of their manifolds, as presented in Section 3.2 (see Figure 3.9). To this end, this section presents an approach that refines the solutions of parallel vectors, by slightly moving and deforming that solution, such that they become as much as possible tangential to the vector field. In particular, to achieve accurate results, the solutions of $\mathbf{u} \parallel \mathbf{a}$ are refined in cases where $\mathbf{u} \parallel \mathbf{b}$ fails.

Figure 3.6a illustrates the refinement algorithm due to a typical solution of $\mathbf{u} \parallel \mathbf{a}$ (blue) when applied for the extraction of curved bifurcation lines (green). Although it solved $\mathbf{u} \parallel \mathbf{a}$, it did not succeed in providing a solution that is close to a streamline, i.e., $\alpha(\boldsymbol{\vartheta}, \mathbf{u})$ is comparably large. Quite large values of τ_α are necessary to obtain a solution at all in such cases. However, the necessity of using large τ_α indicates that the result obtained is of low quality, i.e., substantially displaced from the aimed bifurcation line. In contrast, the



(a)



(b)

Figure 3.7 — (a) Convergence plots of the refinement algorithm. The minimum $\phi_{\min}$, maximum $\phi_{\max}$, and average $\overline{\phi}$ absolute flux of the vector field across the segments of the polylines over the iterations of the refinement. In this case, the refinement of the solution of $\mathbf{u} \parallel \mathbf{a}$ (purple) in (b) converging to the bifurcation line (green in (b)). Zero absolute flux represent lines tangent to the vector fields, i.e., streamlines. (b) Impact of regularization (dataset II): refinement of initial solution of $\mathbf{u} \parallel \mathbf{a}$ without regularization (black) and with (green).

results in Figures 3.5a and 3.5b show for the example of $\mathbf{u} \parallel \mathbf{a}$ (purple) and $\mathbf{u} \parallel \mathbf{b}$ (yellow), that, with the refinement algorithm presented in this section, the bifurcation line is well captured (green) even if $\alpha(\boldsymbol{\vartheta}, \mathbf{u})$ is large.

Hence, the motivation is to refine the initial solutions provided by $\mathbf{u} \parallel \mathbf{a}$ (or $\mathbf{u} \parallel \mathbf{b}$) in order to reduce the angle $\alpha(\boldsymbol{\vartheta}, \mathbf{u})$, i.e., one aims to apply the least necessary deformation (in distance and shape) to make them fit a streamline as close as possible. Martinez Esturo et al. [2013] presented a technique that obtains streamsurfaces by deforming arbitrary surfaces to be as tangential as possible to a vector field with a Poisson-based method. Their approach could serve for streamlines as well, although further investigation would be needed. As presented here, however, for streamlines a much simpler approach can be followed. Notice that, application of Martinez Esturo et al. [2013] would exceed the scope of this chapter. Nevertheless, as future work it would be worthwhile to investigate their approach to this end.

The refinement algorithm consists of an iterative gradient descent approach. At each iteration, the tangent $\boldsymbol{\vartheta}_i$ of the feature polyline at each of its vertices $\mathbf{c}_i$ is estimated by $\boldsymbol{\vartheta}_i := \mathbf{c}_{i+1} - \mathbf{c}_{i-1}$ and a plane normal to that tangent is constructed (orange in Figure 3.6). Note that, this tangent is clamped at the boundaries of non-cyclic feature lines, i.e., it is defined by either $\boldsymbol{\vartheta}_i := \mathbf{c}_i - \mathbf{c}_{i-1}$ or $\boldsymbol{\vartheta}_i := \mathbf{c}_{i+1} - \mathbf{c}_i$ there. Inside this plane the vertices are shifted in the direction that reduces $\alpha(\boldsymbol{\vartheta}_i, \mathbf{u}_i)$, i.e., along $-\nabla\alpha(\boldsymbol{\vartheta}_i, \mathbf{u}_i)$ (red in Figure 3.6), with $\mathbf{u}_i$ representing the velocity at $\mathbf{c}_i$. $\nabla\alpha(\boldsymbol{\vartheta}_i, \mathbf{u}_i)$ is estimated by central differencing of $\alpha(\boldsymbol{\vartheta}_i, \mathbf{u}_i)$ using a four-neighbor stencil in the plane. Because Newton iterations would choose a different step size for each vertex in generic setups and would perturb the tangents, constant step size Δt is hence used for all vertices and Δt is reduced after each iteration: $\Delta t_{new} = \delta \Delta t_{old}$. In the experiments $\delta = 0.999$ was used and Δt was initially set to 0.01 times the cell size. Centering the orange refinement planes at the respective vertices, as Figure 3.6a might suggest, would, however, lead to uneven distribution of the vertices along the polyline in asymmetric configurations, i.e., the polyline segments would tend to vary in size. This would affect the estimation of the tangent and hence make the refinement unstable. To avoid this, a regularization is employed, i.e., the planes are translated along $\boldsymbol{\vartheta}_i$ such that they intersect $\boldsymbol{\vartheta}_i$ at $(\mathbf{c}_{i+1} + \mathbf{c}_{i-1})/2$, as illustrated in Figure 3.6b. This leads to a more uniform distribution of the polyline vertices during refinement and hence better convergence. Note that, the black curve in Figure 3.7b, which is obtained with the refined $\mathbf{u} \parallel \mathbf{a}$ without regularization, is very irregular, discontinuous, and exhibits very high $\alpha(\boldsymbol{\vartheta}, \mathbf{u})$, while the green curve, obtained with refinement of the same solution $\mathbf{u} \parallel \mathbf{a}$ with regularization, correctly captures the bifurcation line.

The absolute flux of $\mathbf{u}$ across the segments of the polyline is computed for evaluation and monitoring of the refinement approach. For such, each segment $\mathbf{s}_i := \mathbf{c}_{i+1} - \mathbf{c}_i$ of the polyline is discretized into n parts of equal length, $\mathbf{u}$ is interpolated at the center of each part resulting in $\mathbf{u}_j$, and the absolute flux ϕ_i across segment i is computed as:

$$\phi_i := \frac{\|\mathbf{s}_i\|}{n} \sum_{j=1}^{n} \left\| \mathbf{u}_j - \frac{\mathbf{u}_j \cdot \mathbf{s}_i}{\|\mathbf{s}_i\|^2} \cdot \mathbf{s}_i \right\| . \tag{3.3}$$

Its minimum $\phi_{\min}$, maximum $\phi_{\max}$, and average $\overline{\phi}$ are measured over all polyline segments at each iteration step of the refinement, and plotted. Figure 3.7a shows the plots due to the refinement of the initial solution of $\mathbf{u} \parallel \mathbf{a}$ (purple) to the bifurcation line (green) in Figure 3.7b. There, the fluxes dropped fast until a local minimum at iteration step 744 (white curve in Figure 3.7b shows the polyline at that iteration), and then slowly to the green curve in 4000 iterations, which took 2.41 seconds (see Table 3.1).

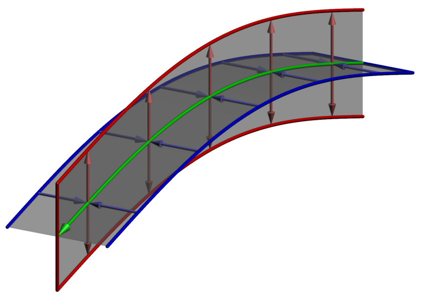


Figure 3.8 — Seed curve generation for 2D manifolds of bifurcation lines. Two seed curves are generated by offsetting from the bifurcation line (green) along the major eigenvector (red) at each point of the line, and two curves by offsetting in the opposite direction of the minor eigenvector (blue). The red curves are connected at the outflow end of the bifurcation line (at the green arrow head), while the blue curves are connected at the inflow end, resulting in two seeding curves.

3.2 Extraction of Bifurcation Manifolds

Each bifurcation line gives rise to a 2D manifold that converges to it in forward time and to a 2D manifold that converges to it in reverse time (see illustration in Figure 3.1). Since a bifurcation line typically represents a streamline, starting a streamsurface from this line would degenerate to a streamline itself. Similar to the extraction of 2D manifolds of saddles-type critical points, one needs to seed the streamsurface at some offset from the bifurcation line. In the case of saddle points, it is common practice to use a closed curve for seeding and to offset this curve from the critical point along the eigenvectors of $\nabla\mathbf{u}$. In analogy to that, a seeding curve for each of the two 2D manifolds at a small offset from the bifurcation line is generated, as illustrated in Figure 3.8. The seeding curves for the manifolds that converge to the bifurcation line in forward time are offset in opposite direction to the minor eigenvector and integrated in reverse time, while those that converge to it in reverse time are offset along the major eigenvector and integrated in forward time.

Interestingly, this straightforward approach is not optimal: it would result in unnecessary gaps, as shown in Figure 3.2c. These gaps are avoided if the forward seeding curves are connected at the outflow end of the bifurcation line and the reverse ones are connected at its inflow end, as illustrated in Figure 3.8. This results in manifolds that are more consistent with the 2D manifolds of saddle-type critical points (see Figure 3.2b). As a consequence, this seeding approach is able to extract periodic orbits by intersecting the two manifolds even in cases where the bifurcation line captures only part of the orbit, such as the periodic orbit in Figure 3.4c.

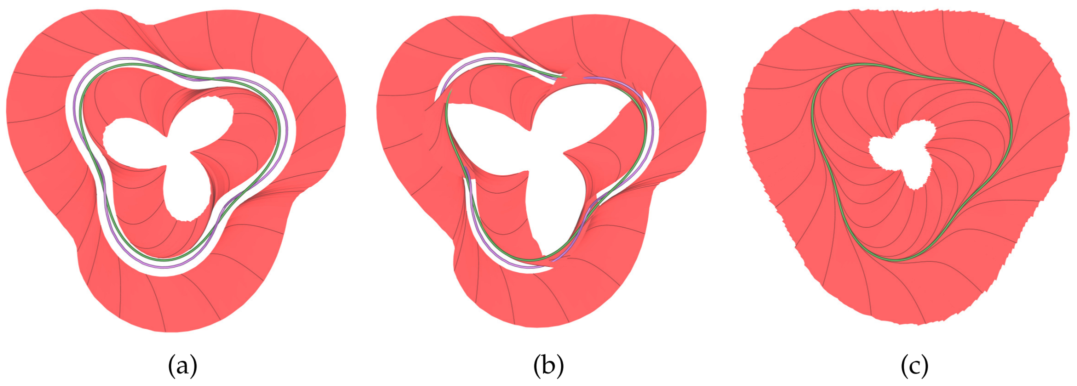


Figure 3.9 — Evaluation of bifurcation lines by smallest possible seeding offset for manifold generation. More accurate (more streamline-like) bifurcation lines allow for smaller seeding offset. The underlying dataset (V) exhibits additional oscillation of the periodic orbit, causing $\mathbf{u} \parallel \mathbf{b}$ to fail. (a) Manifold (red) computed from solution of $\mathbf{u} \parallel \mathbf{a}$ (purple) with manually found minimum offset of 1.5 that still assures that manifolds extend to correct side of bifurcation line (in contrast to (b) where offset was set to 1.0). (c) Same as (a) but manifold computed from refined $\mathbf{u} \parallel \mathbf{a}$ (green) allows for much smaller offset (smaller than 0.05).

As in the case of the extraction of 2D manifolds of saddle-type critical points, where the allowable offset depends on the accuracy of the critical point extraction, the size of the offset is limited by the accuracy of the bifurcation line extraction, too. If the offset is chosen too small, at least parts of the manifolds are advected to the wrong side of the bifurcation line. Hence, the required offset is an expressive measure for the accuracy of an extracted bifurcation line. It can be determined only for the reason of quality assurance, or it is obtained as a byproduct during manifold extraction. Figure 3.9 illustrates this in the context of the evaluation of $\mathbf{u} \parallel \mathbf{a}$ and refined $\mathbf{u} \parallel \mathbf{a}$.

3.3 Results

The approach was evaluated using different vector field datasets. As (incompressible) liquid flow fields at moderate Reynolds numbers exhibit negligible divergence and magnetic fields are divergence-free, critical points as well as periodic orbits of type (spiral) source and sink do not need to be considered in these data. Hence, the remaining stable traditional topological constructs in such 3D vector fields are saddle-type critical points (and the resulting saddle connectors), and (twisted) saddle periodic orbits. As detailed in this chapter, both of these constructs contain bifurcation lines. This bifurcation line extraction is investigated in the field of CFD in Section 3.3.2, as it is a common source of vector fields, and MHD data of the solar corona in Section 3.3.3. But since CFD and MHD datasets containing saddle-type periodic orbits were not successfully found, respective analytic models of orbits were developed, which is covered in Section 3.3.1. Note that, in this chapter, some of the streamlines of the 2D manifolds are highlighted to depict flow direction.

Table 3.1 provides the timings of this approach for the datasets used in the following results. The experiments were performed on a Linux-based system consisting of an Intel(R) Core(TM) i7-2600 Central Processing Unit (CPU) at 3.4 GHz with 8 GiB of Random-Access Memory (RAM) and equipped with a NVidia GeForce GTX 560 Ti with 2 GiB of memory. Note that the prototype implementation was not optimized for performance and there is potential for further acceleration.

3.3.1 Synthetic Periodic Orbits

For investigating the technique in the context of periodic orbits, two similar mathematical models of saddle-type periodic orbits were developed and sampled both on a Cartesian grid of 60^3 nodes and extent of 59^3.

Table 3.1 — Performance measurements. Total includes extraction (Sec. 3.1.1), refinement (Sec. 3.1.2), and filtering (Sec. 3.1.1).

Data	**Time[s]**		
	Refine	**Filter**	**Total**
Buoyant Flow (Fig. 3.11b)	3.393	0.196	9.145
Saddle Orbit I (Figs. 3.4a and 3.5a (green))	0.662	0.371	1.597
Saddle Orbit II (Figs. 3.5b and 3.7b (green))	2.409	0.373	3.343
Saddle Orbit III (Fig. 3.5c (green))	1.076	0.365	2.367
Saddle Orbit IV (Fig. 3.4b)	0.006	0.367	1.320
Saddle Orbit V (Figs. 3.9 (green))	0.436	0.366	1.361
Saddle Orbit VI (Fig. 3.4c)	0.631	0.367	1.514
Saddle Orbit VII (Fig. 3.10 (green))	0.458	0.368	2.989
Solar MHD (Fig. 3.13b)	43.848	14.964	81.303

The first model represents an open vector field (with flux across its domain boundaries) and exhibits a saddle-type periodic orbit while the hyperbolic influence is not spatially bounded, i.e., it extends throughout the domain. Besides the definition of the radius of the orbit, it provides parameters for the radial and axial linear growth of the respective vector components with respect to the distance from the orbit, i.e., the hyperbolicity, a parameter for the speed of the superimposed rotation along the orbit, amplitude and frequency to add a radial sine oscillation of the orbit, and amplitude and frequency to add an axial sine oscillation of the orbit.

The second model represents a spatially bounded 2D hyperbolic region, oriented perpendicular to the orbit, bounded with a cosine profile and moving along the orbit. It provides parameters, additional to the first model, to set the speed of twist (i.e., rotation around the orbit) of the hyperbolic region as it moves along the orbit, parameters to define the size of the hyperbolic region, and to vary the hyperbolic strength of the region while it moves along the orbit.

The first model gave rise to the following datasets. The dataset I, visualized in Figures 3.4a and 3.5a, exhibits low rotation (0.4) along the orbit. The dataset II from Figures 3.5b and 3.7b differs from that only in terms of rotation along the orbit, which is 1.3 in this case. In the dataset III from Figure 3.5c, the rotation is 3, and the dataset IV of Figure 3.4b differs once more only in terms of the rotation along the orbit, which is 10.0 there. One exception is the dataset V from Figure 3.9 where the hyperbolic component in axial direction is smaller, rotation "along" the orbit is only 0.3, and the orbit oscillates radially with frequency 3 and amplitude 3 and axially with frequency 4 and amplitude 0.8 per revolution.

The second model was used to generate the remaining two datasets. The dataset VI, visualized in Figure 3.4c, exhibits rotation of 0.05 along the orbit and the hyperbolicity varies from maximum on one side of the orbit to zero hyperbolicity on the other side. Dataset VII (Figure 3.10) is the most complex one, it differs from VI in terms of slower rotation along the orbit (0.02), a 180 degree rotation around the orbit per revolution (causing a twisted-saddle periodic orbit), no variation of hyperbolicity along the orbit, but instead radial oscillation of the orbit with amplitude 2 and frequency 6, axial oscillation with amplitude 3 and frequency 5, and noise of maximum magnitude 0.005 added to the vector field. Adding higher levels of noise, however, disrupted the results of $\mathbf{u} \parallel \mathbf{a}$. A more sophisticated noise model could be used for this. This data is considered representative for real-world data.

The technique obtained the periodic orbit in all cases in terms of a bifurcation line, some with refined $\mathbf{u} \parallel \mathbf{a}$, some with $\mathbf{u} \parallel \mathbf{b}$. As expected, however, it extracted only the sufficiently hyperbolic part of the orbit in dataset VI. The 2D manifolds of the bifurcation line, however, obtained the complete periodic orbit due to the seed curve connection approach (Figure 3.8). Note that because $\mathbf{u} \parallel \mathbf{a}$ provided a very inaccurate result in this case, the ends of the resulting line did not converge well to the bifurcation line during refinement.

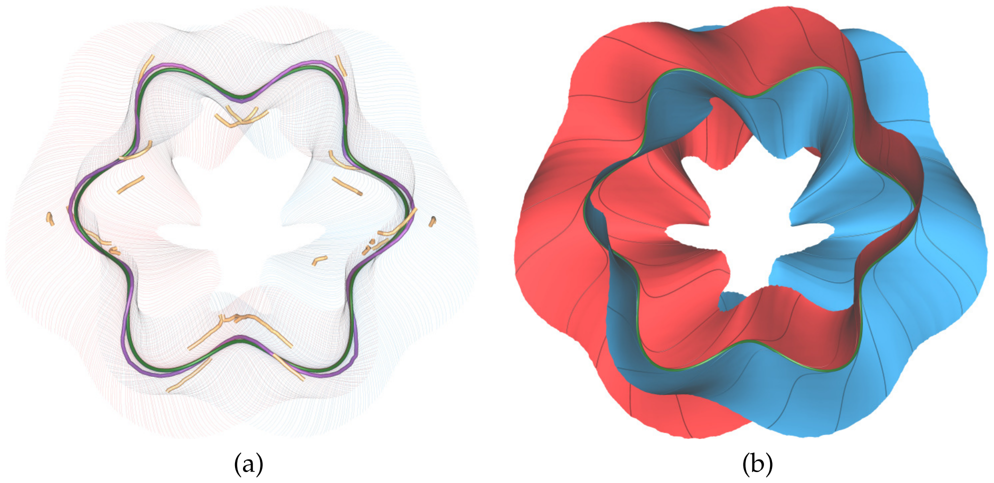
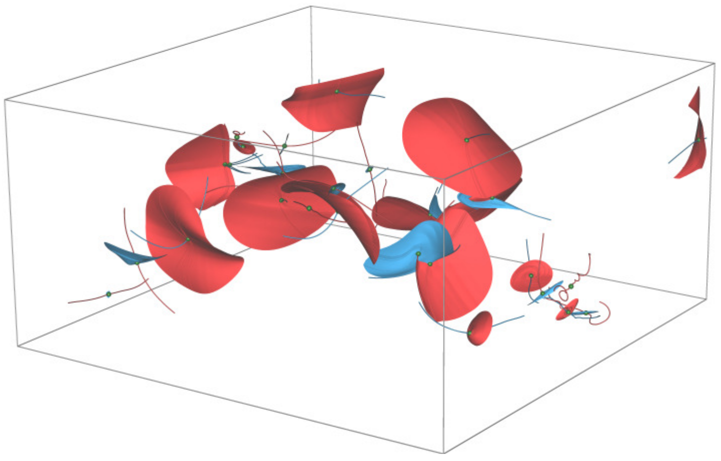


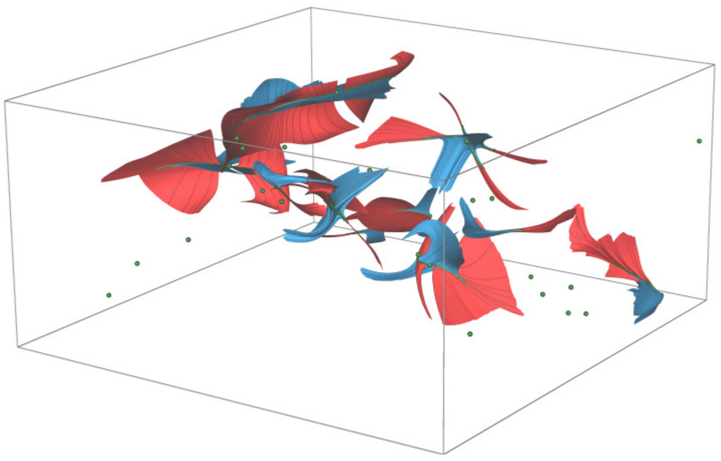
Figure 3.10 — Twisted Saddle Orbit dataset (VII). Similar to 3.4a but with orbit additionally curved in both radial and axial direction, and with noise added to the vector field. (a) Parallel vectors solutions of $\mathbf{u} \parallel \mathbf{a}$ (purple), refined $\mathbf{u} \parallel \mathbf{a}$ (green), and $\mathbf{u} \parallel \mathbf{b}$ (yellow), with some of the streamlines of the manifolds for context. The $\mathbf{u} \parallel \mathbf{b}$ criterion failed here, as in most curved periodic orbits with low tangential velocity in the experiments. (b) The manifolds of refined $\mathbf{u} \parallel \mathbf{a}$.

3.3.2 Buoyant Flow CFD Data

Next, this approach was applied to a time-dependent simulation of buoyant air flow in a closed container, heated at the bottom and cooled at the top. This dataset was simulated on a uniform grid with resolution $61 \times 31 \times 61$. Figure 3.11a shows the result obtained with traditional vector field topology while Figure 3.11b shows the result of the visualization with bifurcation lines and its manifolds according to this extraction approach. One can see that the 2D manifolds extracted from both approaches are consistent, although some are missing in one approach while some are missing in the other. For example, the unstable 2D manifold of the uppermost saddle point was also obtained from the bifurcation line. Nevertheless, at the same region, the stable 2D manifold could only be extracted from the bifurcation line. On the other hand, there are cases where a critical point provides manifolds that are not related to bifurcation lines, e.g., those of spiral saddles at the bottom. This demonstrates that a merge of both approaches provides a more complete visualization of the topological structure of vector fields.



(a)



(b)

Figure 3.11 — Buoyant Flow dataset. (a) Traditional visualization by means of 1D/2D manifolds (red, blue) of (spiral) saddle-type critical points (green). (b) Visualization by means of 2D manifolds (red, blue) of bifurcation lines (green curves) provides insight in cases where only one or no critical point at all is involved. Note that in (a) there are also 2D manifolds not related to bifurcation lines (e.g., those of spiral saddles at bottom right), but since those manifolds that are present in both (a) and (b) are consistent with each other, the union of both provides the full topological structure. See Figure 3.12 for a closeup on the critical point at the top.

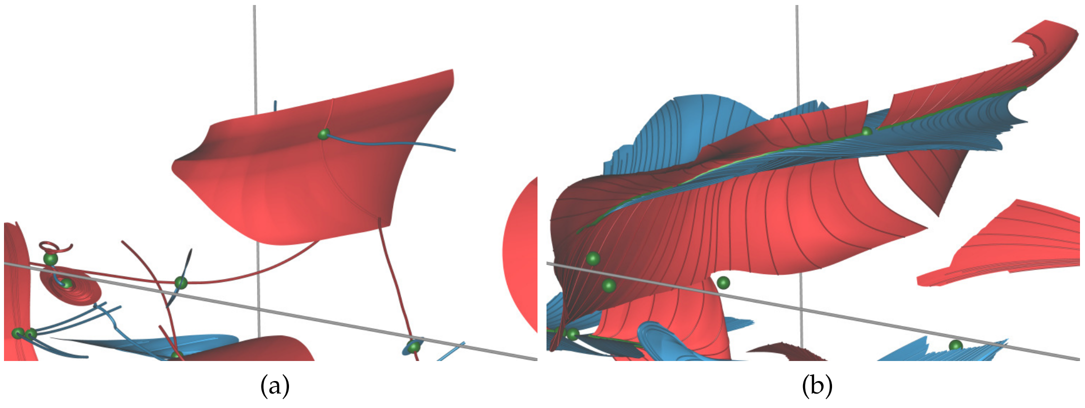
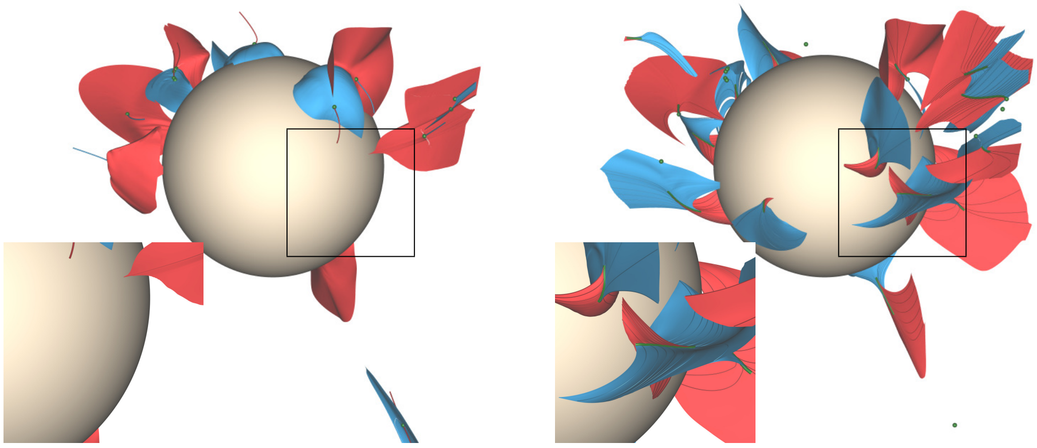


Figure 3.12 — Closeup on critical point at the top region of Figure 3.11. (a) Traditional visualization with 1D/2D manifolds (red, blue) of saddle-type critical points (green) and (b) visualization with 2D manifolds (red, blue) of bifurcation lines (green curves). One can see that the blue manifolds from bifurcation lines provide more insights on the topology than with the traditional approach and that the red manifolds from the bifurcation lines are consistent with the result from the traditional approach.

3.3.3 Solar MHD Data

Finally, experiments were carried out with MHD simulation data of the solar corona, available at the *Predictive Science Modeling Support for Helioseismic and Magnetic Imager Solar Dynamics Observatory* [pre]. These data are given on a spherical structured grid at resolution 181 × 100 × 150, but since the coronal magnetic field exhibits significantly more complex structures near the solar surface, the data was resampled in regions of interest (ROI) there.

The first investigation is in a small ROI that contains two saddle points close to each other. Figure 3.2a shows the result by means of traditional vector field topology: both critical points have been extracted and the respective 1D and 2D manifolds computed. It can be seen that both 2D manifolds converge to and are limited by the 1D manifold of the other critical point. The intersection of the two 2D manifolds represents a saddle connector. The streamlines on the 2D manifolds nicely convey that the complete saddle connector is hyperbolic, i.e., it repels streamlines in one direction along one 2D manifold while it attracts streamlines along the other 2D manifold. Figure 3.2b provides the result obtained with bifurcation lines. Because the entire saddle connector is hyperbolic, it is completely captured by the resulting bifurcation line. Note that there are also small bifurcation lines on the other side of the critical points, resulting in gaps in the 2D manifolds where the critical points reside. These



(a) (b)

Figure 3.13 — Solar MHD dataset. (a) Traditional visualization by means of 1D/2D manifolds (red, blue) of (spiral) saddle critical points (green). (b) Visualization based on bifurcation lines (green curves) and their 2D manifolds (red, blue). It is apparent that (a) fails to provide the field structure in regions where no critical points are present (e.g., the front of the Sun), while the visualization with bifurcation lines provides these details.

gaps could be closed by connecting the bifurcation lines across the critical points prior to seed curve generation, but this was omitted in the implementation as it could introduce inaccuracies.

Another ROI captures a bifurcation line that extends from a saddle point to the domain boundary (Figure 3.3). In this case, traditional vector field topology only provides one 2D manifold, see Figure 3.3a. Figure 3.3b shows the result obtained with bifurcation lines due to this approach, which provided both 2D manifolds and hence the full topological picture. Note that the 2D manifolds are not integrated to full extent to avoid occlusion issues.

Figure 3.13 compares the result from traditional vector field topology with bifurcation lines. This ROI was resampled at a resolution of 201^3. As for the CFD dataset, one can observe that in some regions (e.g., the highlighted one) the traditional vector field topology does not provide the full topological picture, i.e., some separating manifolds are missing, while the visualization with bifurcation lines provides those and hence complements it.

Chapter 4

Solar Dynamics

When observed at low spatial resolution in the visual part of the electromagnetic spectrum, the Sun appears as a rather unspectacular, quiescent, bright disk, and it is only during solar eclipses that details of its extended outer "atmosphere", called *corona*, become observable from Earth. Therefore, one of the earliest strategies to visualize the corona is with a coronagraph, which consists of an equipment that produces an artificial eclipse by positioning a disk-shaped material that occults the Sun. However, in the ultraviolet and X-ray regime of the spectrum, our Sun reveals its spectacular dynamic nature and the energetic processes in its corona and *photosphere* (i.e., the Sun's surface). Most of the solar phenomena are directly related to magnetic fields. For example, *coronal holes* are typically found at the solar poles and consists of those regions at the photosphere from where the magnetic field lines escape into space. From there, the fast component of the solar wind is discharged. The solar wind, which consists mainly of charged particles released from the upper corona, continuously impinges upon the magnetosphere (i.e., the Earth's magnetic field) defining its global shape. At the photosphere, an Active Region (AR) is a region with an exceptionally strong magnetic activity and they are related to many further phenomena. Sunspots often appear in AR and are caused by the Sun's magnetic field emerging at the photosphere and revealing its inner material that exhibits lower temperatures, and are therefore observed as dark spots. Coronal loops are currents of plasma typically at the lower corona traveling along the coronal magnetic field lines that emerge and later on return to the photosphere. Moreover, single energetic events like solar flares, which consist of sudden explosions at the Sun, also form in active regions and are related to a phenomenon called Coronal Mass Ejection (CME) that releases solar matter above the corona interacting with the solar wind. A CME is called Interplanetary Coronal Mass Ejection (ICME) if the matter travels further possibly reaching planets. If an ICME conveys towards the Earth, it can cause geomagnetic storms (i.e., disturbances in the magnetosphere) and can damage, for example, power grids—such as happened in March 1989, when a geomagnetic storm as a consequence of an ICME caused a massive blackout in Quebec/Canada—or communication satellites. A solar eruption as powerful as the one of September 1859, which caused the largest geomagnetic storms on record, is likely to happen in the next years and could cause complete power blackout. Hence, there is an increased interest in monitoring—and ultimately forecasting—the Sun's activity and the resulting space weather phenomena not only from a scientific point of view but also from a commercial one.

Currently, some of the main scientific objectives in solar research are: How does the Sun generate its magnetic field? How does that field produce flares and CME? How is the solar corona being heated? And how does the Sun influence space weather? On the way to help answering these fundamental astrophysical questions, this work and the resulting future work in solar visualization will contribute to new insights, new techniques, and eventually new models.

On February 11th, 2010, the National Aeronautics and Space Administration (NASA) launched the Solar Dynamics Observatory (SDO), which is the first space mission as part of the project Living With a Star (LWS) that aims to monitor and study the Sun and its impacts on Earth. The SDO is on a geosynchronous orbit around the Earth and has three main instruments onboard—Helioseismic and Magnetic Imager (HMI), Atmospheric Imaging Assembly (AIA), and Extreme Ultraviolet Variability Experiment (EVE)—that captures at an almost continuous time cadence high-resolution solar images of up to 4096^2 pixels, which corresponds to $1''$ on the *photosphere* (i.e., the Sun's surface), and at multiple wavelengths, which results in terabytes of raw image data every day. While a solution for browsing this huge image archive is already realized by the JHelioviewer project [Müller et al., 2009; jhe] from European Space Agency (ESA), there is no tool for an interactive exploration of more complex data products related to physical quantities that are derived from the raw image data. This, however, is indispensable for a deeper understanding of the magnetized plasma that constitutes the solar atmosphere.

This chapter applies visualization techniques ranging from volume visualization over flow visualization to space-time visualization, making use of the focus+context and linked views elements. The utility of the proposed approaches through selected problems in solar physics are exemplified in joint work with two domain experts. Their evaluations and physical backup are presented in Section 4.4.1.

This chapter presents the following contributions to the field of solar dynamics data visualization:

- raycasting for solar data in spherical grids,
- 3D LIC as an alternative for field line placement,
- a data-driven LIC-based technique for visualizing the magnetic field, imitating the self-illustrating phenomenon of coronal loops,
- an interactive view-dependent visualization of open field regions on the photosphere, i.e., *coronal holes*, and
- space-time visualization of photosphere data.

4.1 Related Work

Visualization looks back on a long history in astronomical research—from the conversion of sensor data to images, to the analysis of simulation results. However, while many of the involved problems are 2D and comparably well supported by appropriate visualization techniques—it is in particular the 3D structure of the Sun's magnetic field that requires visualization approaches beyond those traditionally used. This topic is gaining importance since the simulation of the 3D magnetic field of the solar corona can be driven by measurements of the magnetic field at the photosphere.

Traditionally, the Sun's magnetic field is visualized by field line placement (Section 2.3), which typically suffers from the influence of the usually randomly, regularly, or manually chosen seed points. Beyond that, some works address the visualization of magnetic fields in general. Sundquist [2003] presents dynamic line integral convolution, that visualizes the movement of magnetic field lines with LIC, Klein and Ertl [2004] present an approach based on a particle model to visualize magnetic field lines, and Thomaszewski et al. [2008] introduce magnetic interaction in rigid body simulations based on dipoles. Sanderson et al. [2010] visualize magnetic fields in fusion reactors, Sadlo et al. [2004] apply magnetic visualization to vortices, and Bachthaler et al. [2012] visualize flux topology of 2D magnetic fields induced by a set of dipoles.

Coronal holes are the solar phenomena most closely related to this overall thesis, as their extraction with respect to the coronal magnetic field consists of mapping the transport of magnetic field lines between two boundaries of the domain, i.e., the photosphere and an escape boundary, typically located at 2.5

times the solar radius. A review of coronal holes and a historical overview is given by Cranmer [2009]. There are some techniques to detect coronal holes. A quite recent technique for automatic detection of coronal holes based on image segmentation is described by Krista and Gallagher [2009]. Riley et al. [2006] use the typical approach for extraction, that precomputes a map by field line integration starting from the photosphere to detect coronal holes. Cai et al. [2005] applied traditional vector field topology to the Earth's magnetotail. Nevertheless, all these techniques have in common that they cannot achieve sufficient resolution to robustly resolve very thin coronal holes.

The JHelioviewer project [jhe] presented by Müller et al. [2009] consists of a Java-based visualization software for browsing large solar image archives using JPEG 2000 (JP2) for compression and efficient random sampling, e.g., of a Region of Interest (ROI), and JPEG 2000 Interactive Protocol (JPIP) to stream data over the Internet. It also provides, among other features, time navigation, image composition from different wavelengths, and different coloring options with predefined transfer functions. SolarSoftWare (SSW) [ssw] is a system based on the Interactive Data Language (IDL), which provides a common programming and data analysis environment for solar physics. In SSW, DeRosa developed a Potential-Field Source-Surface (PFSS) viewer to show magnetic field lines based on line-of-sight magnetograms from the Michelson Doppler Imager (MDI) of the Solar and Heliospheric Observatory (SOHO). MDI is the precursor of the HMI onboard the SDO, both developed at the University of Stanford.

First models of the solar corona's global magnetic structure were developed by Schatten et al. [1969] and Altschuler and Newkirk [1969]. They extrapolated the radial component of the magnetic field measured in the photosphere using a potential field model where electric currents are neglected and the field is forced to be radial at 1.5 solar radii above the surface. Although this is a very crude simplification, it delivers valuable results in a wide range of solar and heliospheric topics and hence it is still in use. Another extrapolation method beside PFSS is the force-free field method. Schrijver et al. [2006], for example, imply that electric currents run only along field lines. Depending on whether the ratio of field strength to current density is constant throughout a volume or not, one refers to linear or Nonlinear Force Free Field (NLFFF) approximation. Wiegelmann et al. [2012] use NLFFF for the extrapolation of magnetic fields in isolated ARs and compare the resulting magnetic field lines with 171 Å SDO/AIA images. Better extrapolations for the coronal magnetic field require sophisticated MHD simulations. While these simulations can also include the solar wind and magnetic reconnection, they need massive computational resources even for small domains. A comparative study of PFSS and MHD models were conducted by Riley et al. [2006].

In this chapter, the PFSS model is mainly used because it is easy to implement on the GPU and it delivers a good first guess of the magnetic fields in a reasonable time (e.g., in the order of minutes).

4.2 Data Sources

The Joint Science Operations Center (JSOC) maintains a system that provides data products from the SDO—among data from other missions, e.g., MDI—in the Flexible Image Transport System (FITS) image format [Pence et al., 2010].

In solar observations, it is common to give the observation time as well as the location of the observer in Carrington Coordinates (CC) (θ_{CR}, ϕ_{CR}), where θ_{CR} is the Carrington Latitude (CRLT) and ϕ_{CR} is the Carrington Longitude (CRLN) (see [Thompson, 2006] for more details on heliographic coordinates). These coordinates "rotate with the sun" and a new Carrington Rotation (CR) defines the reference meridian of the Carrington longitude.

The Solar Region Summary (SRS) is a daily report of ARs compiled by experts from the Space Weather Prediction Center (SWPC). It contains the location of sunspot groups in heliographic coordinates as well as its area size and a classification.

The AIA instrument filters cover 10 different wavelength bands in the ultraviolet and extreme ultraviolet spectrum and map different regions of the solar atmosphere from the photosphere up to the corona. In this chapter, the 171 Å filter that is sensitive to the Fe IX line at a temperature of about 630,000 Kelvin is of particular interest because it shows coronal loops at the best, as one can see in Figure 4.1a.

The HMI instrument determines the magnetic field at the photosphere by measuring the spectral profile of the Fe I absorption line at 6173 Å (see Figure 4.1b). In this chapter, HMI magnetograms, which yield the line-of-sight component of the magnetic field vector in the photosphere for various purposes, and the HMI continuum filtergrams, which provide broad-wavelength photographs of the photosphere for our 'plasma flux' texture, presented in Section 4.3.5, are used. Both, the AIA and HMI data are resampled using supersampling on demand in regions of interest in LCEA coordinates (see below).

The Global Oscillation Network Group (GONG) [gon] is a network of six earth-bound instruments measuring the solar magnetic field and oscillations on the solar surface. The line-of-sight component of the magnetic field on the solar surface is publicly available in the form of synoptic 360° (non-synchronic) magnetograms in 360×180 pixels resolution prepared using Lambertian-Cylindrical-Equal-Area (LCEA) projection (see Figure 4.2). This projection maps latitude θ

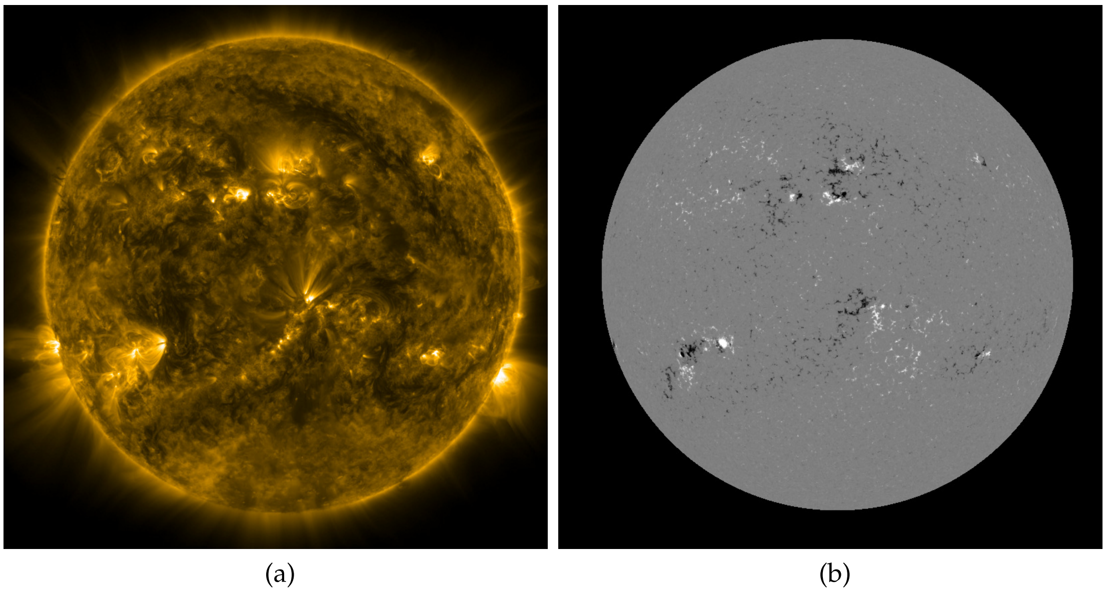
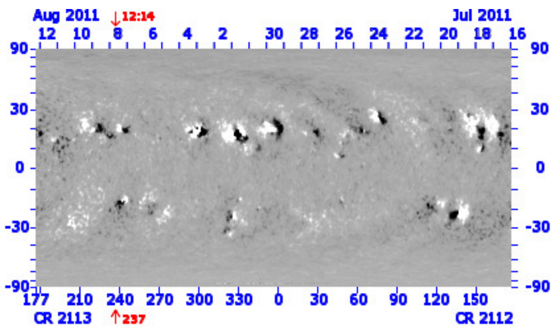


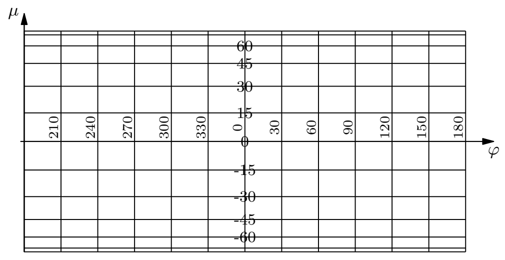
Figure 4.1 — AIA 171 Å image (a) and HMI magnetogram (b) @ 2012.03.27, 02:02Z. (Courtesy of NASA/SDO and the AIA science team.)

and longitude ϕ to Cartesian coordinates (x, y) via $x = \phi - \phi_0$ and $y = \sin\theta$ with ϕ_0 being the central meridian. Synoptic magnetograms are constructed from helioprojective-Cartesian coordinates by unrolling the central meridian over time. As it delivers data for the whole solar surface, it is used to prepare a Spherical Harmonics (SH) decomposition. The GONG website provides the synoptic magnetograms in the FITS format—alternatively also in the format by the Joint Photographic Experts Group (JPEG)—and the SH coefficients up to order $n = 40$ with a cadence of roughly every hour.

The most realistic approach to provide 3D data of the Sun's magnetic field makes use of a full 3D global MHD model [Mackay and Yeates, 2012]. The Magnetohydrodynamics Around a Sphere (MAS) system provides precomputed datasets with coronal magnetic fields given in spherical coordinates that can be downloaded from the *Predictive Science Modeling Support for Helioseismic and Magnetic Imager Solar Dynamics Observatory* [pre]. In one of the central parts of this thesis, specifically in the visualization of coronal holes (Section 4.3.7) and escape maps (Chapter 5), the MHD data from [pre] are used. However, the present chapter mainly uses spherical harmonics for a global representation and the synoptic magnetograms as "space-time atlas" for overview and for selections of interesting regions.



(a)



(b)

Figure 4.2 — GONG synoptic magnetogram of 2011.08.08,12:14Z with initial Carrington longitude CRLN $= 177°$ (a). LCEA coordinate lattice (b).

4.2.1 Coronal Magnetic Field Extrapolation on GPU

While the magnetic field can readily be measured spectroscopically at the solar photosphere, this is unfortunately not feasible in the solar corona due to its much lower density. For this reason, extrapolation methods are used to reconstruct the coronal magnetic field based on photospheric data. The simplest one—the PFSS model—was developed by Altschuler and Newkirk [1969] and Schatten et al. [1969]. It assumes that the coronal magnetic field is current-free, $\nabla \times \mathbf{B} = \mathbf{0}$, and thus can be represented by the gradient of a scalar potential, $\mathbf{B} = \nabla\psi$. Together with the divergence-free condition of a magnetic field, $\nabla \cdot \mathbf{B} = 0$, the scalar potential has to fulfill the Laplacian equation $\nabla^2\psi = 0$. The boundary conditions are defined on the solar surface, $R_\odot = 1$, by the magnetogram, and at the radius $r = R_w = 2.5R_\odot$ by $\psi_r = 0$, where the field is assumed to be purely radial.

The solution to the Laplacian in spherical coordinates is usually expanded into spherical harmonics [Altschuler and Newkirk, 1969]. Because direct evaluation of the spherical harmonics is computationally expensive, the data is resampled on a spherical LCEA grid in a preprocessing step. Please note that spherical LCEA coordinates (r, μ, φ) are used where $\mu = \sin\theta = \cos\vartheta$ with latitude θ instead of colatitude ϑ as used in traditional spherical coordinates.

As pointed out by Tóth et al. [2011], the traditional spherical harmonics decomposition works reasonably well when the order of spherical harmonics is limited to be small relative to the resolution of the magnetogram. However, around sharp features, ringing artifacts become inevitable. Furthermore, spherical harmonics are global functions and their amplitudes depend on all of the magnetogram data. Since the basis of the spherical harmonics are the synoptic magnetograms, the global magnetic field is influenced by data that might have already changed significantly.

The finite difference method for PFSS has the advantage to be applicable also for local domains. In this chapter, after interactively selecting a domain on the visible side of the Sun, a regular curvilinear grid bounded by either $\mu = \text{const}$, $\varphi = \text{const}$, or $r = \text{const}$ is defined. The lower boundary condition, at $r = R_\odot = 1$, is defined by the line-of-sight HMI magnetogram which is resampled for this grid. The boundary conditions at the other five faces follow from the spherical harmonics model using the coefficients by GONG. Then, the Laplacian equation $\nabla^2\psi = 0$ yields a sparse linear system which is solved by the Krylov-type iterative method BiCGSTAB from the Cusp-library [Bell and Garland, 2012] on the GPU. The technical details for coronal magnetic field extrapolation on the GPU can be found in Appendix A.

4.3 Visualization

The first stage of this visualization application provides the SDO data on a 3D sphere representation (Section 4.3.1) and focuses on the investigation of the 3D magnetic field from GONG spherical harmonics data or derived by the PFSS approach (Section 4.2.1), possibly obtained in an interactive manner (Section 4.3.6). As all are provided on spherical LCEA grids, the visualization techniques have to take this into account. To achieve interactive response times and ease implementation, it is operated in Computational Space (c-space) on the GPU where applicable, including c-space field line integration (Section 4.3.2), c-space raycasting (Section 4.3.3), c-space line integral convolution (Section 4.3.4), and a variant thereof for visualization inspired by coronal loops (Section 4.3.5).

The second part (Section 4.3.8), is realized as a linked-view and focuses on the spatiotemporal visualization of sensor data confined to the photosphere, in particular the HMI magnetograms. Volume rendering is applied to reveal the space-time structure in these data and provide space-time curves of the feature classifications present as SRS data (Section 4.2) for context. This approach outperforms traditional temporal visualization by simple playback of the 2D SDO streams.

4.3.1 Solar Sphere Representation

The basis of the spatial visualization view is a spherical model of the Sun, representing the photosphere at solar radius $R_\odot = 1$. While traditional visualization tools for solar data operate in 2D plus time, this work provides new possibilities and context by bringing the data from AIA, HMI, and the synoptic maps from GONG onto the solar sphere with the use of a simple raytracing approach, as

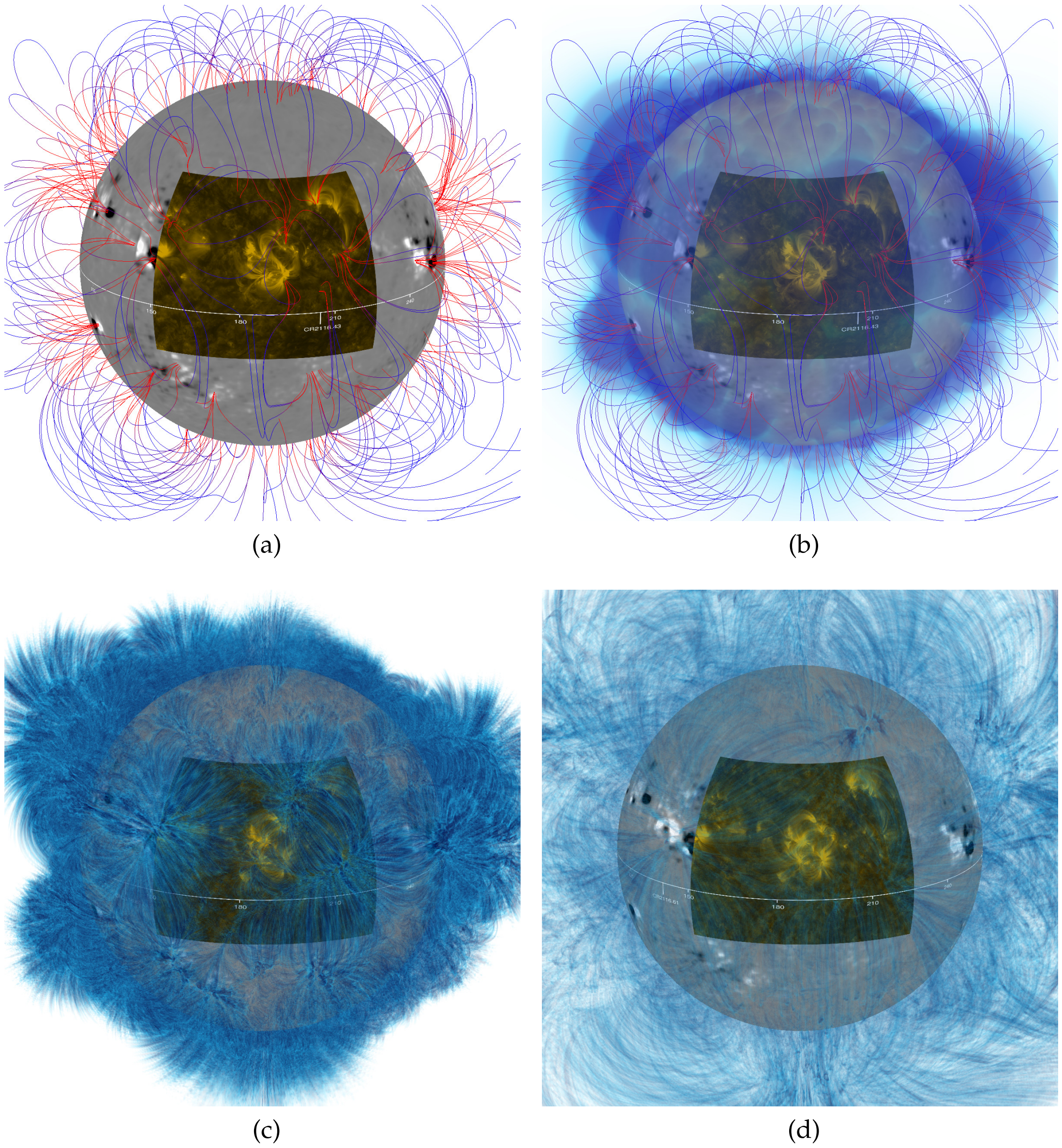


Figure 4.3 — (a) Solar sphere representation with GONG synoptic data (gray) and selected region of AIA 171 Å. Visualizations of GONG SH data with field lines colored with field magnitude and additional volume rendering of magnitude (b), LIC with opacity proportional to magnitude (d), and LIC without opacity variation by field magnitude (c).

illustrated in Figure 4.3a. The main parametrization of this visualization are the Carrington coordinates (CC) (Section 4.2). CRLN, CRLT, and CR are plotted on the sphere, together with an indicator showing the Earth's relative position, i.e., the current CR. As the CC system includes both a rigid body approximation of solar rotation and the motion of the Earth around the Sun, it is independent of these two mechanisms—solar features, such as AR, exhibit small motion in this system.

4.3.2 Field Line Integration

The solar sphere representation is particularly well-suited for integration with other 3D visualization techniques such as field lines (see Figure 4.3a) and volume rendering. While field line integration inside uniform grids is straightforward, retrieving them from spherical grids requires some additional effort. Spherical grids can be unstructured, which would involve algorithmic and storage overhead due to irregular topology, and would particularly make the point location procedure difficult and hard to parallelize on GPUs. However, since the field discretization is derived in this application from GONG spherical harmonics or using PFSS, there is the flexibility to choose a grid type that is more appropriate for processing on GPUs. For this purpose structured curvilinear grids are used, with spherical geometry (see Figure 4.4a). Hence, operations that can be performed in the c-space of these grids (see Figure 4.4b), become straightforward and efficient on GPUs.

To solve the line integration problem in c-space, one has to transform the vectors accordingly, i.e., adapt their magnitude and direction with respect to the geometry of the grid. If the shape of the grid is given by a function $\mathbf{\Gamma}(\xi)$ that maps a position ξ from c-space to Physical Space (p-space), the vectors would have to be transformed from p-space to c-space by multiplication with the matrix $(\nabla\mathbf{\Gamma}(\xi))^{-1}$. This approach, however, would have two drawbacks: the difficulty of gradient estimation in curvilinear grids, and matrix inversion. In this work, both problems can be avoided because in case of SH or PFSS data, the vector field $\mathbf{B}$ is obtained as the gradient of the scalar potential ψ,

$$\mathbf{B} = -\nabla_x \psi = -\frac{\partial \xi}{\partial x} \nabla_\xi \psi, \tag{4.1}$$

where $\partial\xi/\partial x$ is the Jacobian between LCEA- and Cartesian coordinates. Hence, it simply computes the gradient only in c-space, $\boldsymbol{\beta} = -\nabla_\xi \psi$, and stores only $\boldsymbol{\beta}$, not $\mathbf{B}$. It can be easily shown that the involved trilinear interpolation of $\boldsymbol{\beta}$ at ξ is identical to trilinear interpolation of $\mathbf{B}$ at $\mathbf{x}$.

4.3.3 Raycasting

Although, the coronal magnetic field is traditionally visualized in terms of selected field lines, their results highly depend on the selection of their seeds.

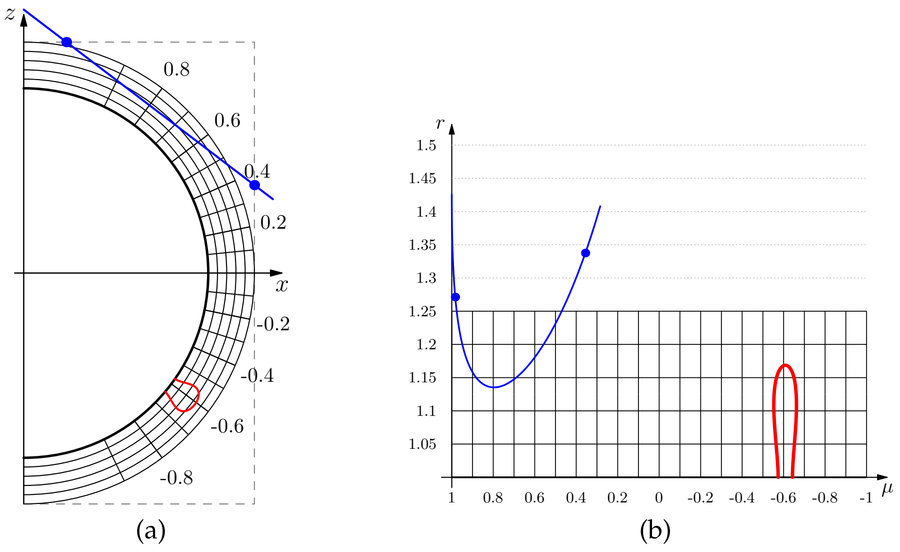


Figure 4.4 — Spherical curvilinear grid adjacent to photosphere (bold line). In p-space the longitude coordinate φ rotates around the z-axis (a). In c-space φ points into the drawing plane (b). Raycasting (blue) in p-space (a) would step along straight lines but along curved lines in c-space (b). Integration is employed along ray inside bounding box in p-space (dotted). Field line integration (red) in c-space allows for trivial and efficient point location. Respective vector field is obtained by gradient estimation in c-space.

By introducing volume rendering related techniques in this field, this work aims at providing better tools for exploration and comparison. To avoid resampling, raycasting also in c-space of the spherical grid is performed. The viewing rays which are straight lines in p-space represent deformed curves in c-space (see Figure 4.4). Although it is simple to derive a formulation of these curves in terms of c-space line integration, it would be rather costly to detect the seeds for integrating those lines, i.e., the points where the rays enter the spherical grid, in particular when only parts of the photosphere are covered by the spherical grid. Instead, in this work, it was decided to stick to the p-space regarding traversal of the rays: standard raycasting code that marches along straight rays in p-space is used, limited by an axis-aligned bounding box of the possibly small spherical grid (see Figure 4.4a). Only for obtaining the value of the field at the respective sample position on the ray, the sample position to c-space is transformed. Figure 4.3b shows a resulting volume rendering of the field magnitude.

4.3.4 Line Integral Convolution

While straightforward 3D LIC often suffers from massive occlusion and visual clutter, it can provide powerful visualizations when constrained to special regions and enriched with alternative geometrical representations [Rezk-Salama et al., 1999]. In this application, 3D LIC is implemented with sparse noise textures and typically with the opacity proportional to the field magnitude (see Figure 4.3c). This readily visualizes the strong magnetic structures.

Computational space LIC Forssell [1994] is achieved in 3D by simply combining the c-space line integration described in Section 4.3.2 with the c-space raycasting from Section 4.3.3. Having the magnetic field $\boldsymbol{\beta}$ in c-space, two additional textures (one noise and one output) are simply created in a user-defined region of interest (ROI) and at user-defined resolution. Then, c-space integration is performed in forward and backward directions starting at each cell of the output texture, the convolution is applied with the values from the noise and the result is stored in a scalar field output texture. The resulting texture is then raycasted in c-space, , see Figure 4.3d. To avoid clutter and to better visualize the physical relevance of $\mathbf{B}$, this resulting texture is typically rendered with opacity proportional to $|\mathbf{B}|$. Note that this requires the field magnitude in p-space, not in c-space. For this, we transform $\boldsymbol{\beta}$ to $\mathbf{B}$ by Equation 4.1.

4.3.5 Virtual Coronal Loops

Particularly well seen in the spectral band around 171 Å (see Figure 4.5d), the coronal loops are the building blocks of the bright X-ray and UV solar corona and subject of particularly active research. They serve as a proxy to study the time-dependent topology of the Sun's magnetic field, which is most relevant, e.g., for constraining models of solar flares and coronal mass ejections. They consist of optically thin and largely transparent plasma tracing out magnetic field lines. At the same time they undergo rapid fluctuations in intensity and shape due to plasma flows, wave phenomena, and thermodynamic fluctuations of the plasma that directly affect its emissivity. These properties make coronal loops a prominent candidate for inspection based on volume rendering techniques. However, 3D models of the global coronal plasma density are not available due to the difficulty of reconstructing the density of the optically thin, low-emission solar corona from remote-sensing observations.

The virtual coronal loops therefore is presented as a technique to support studies of the linkage between magnetic structures observed in the Sun's photosphere and the modeled 3D magnetic field of the corona. In particular, it shall give long-term insight to which extent modern visualization techniques can be used to reproduce the appearance of coronal loops observed in AIA's 171 Å extreme ultraviolet spectral band using only data acquired in the photosphere as input. This technique consists of constructing a 2D texture based on a virtual plasma 'flux' at the photosphere that, combined with LIC, which traces out magnetic field lines, imitates the self-illustrating phenomenon of coronal loops.

The correlation between fine magnetic structures in the HMI magnetograms and the loops in AIA 171 Å is apparent from SDO image comparison. However, detailed inspection combined with HMI continuum data reveals that this correlation does not hold at sunspots. Thus, for the generation of a virtual 'plasma flux' texture, we clamp the magnitude of the HMI magnetogram and set it to zero using a sunspot mask built from the HMI continuum by a simple threshold operation.

The problem of plasma distribution is addressed along the magnetic field using LIC, instead of more involved and computationally more demanding advection schemes that would be able to assure mass conservation. Instead of using standard LIC with a 3D noise texture, the 'plasma flux' texture is used as 2D noise texture and this texture is located at the photosphere (see Figure 4.6). Note that this application operates in c-space, where the photosphere is planar and located at the bottom side of the grid. Specifically, LIC is performed with maximum integral curve length in both directions from each Volume Element (voxel), and it is detected if these lines intersect the 2D texture. If they do so, the values at the intersection points are looked up and the resulting LIC value is obtained by averaging them. Otherwise the resulting LIC value is zero. This scheme distributes the values from the 2D texture in 3D along the field lines (see Figure 4.6). During rendering of the resulting LIC field, the opacity is set proportional to the field magnitude, which, due to Gauss' theorem in flux tubes and the absence of divergence in magnetic fields, indirectly accounts for mass conservation. Note that it is beneficial to generate the 2D texture at higher resolution than the LIC field because this provides more detailed visualization, in particular in regions of divergent field lines. Figures 4.5 (b) and (c) shows an example using PFSS data within a ROI, and Figure 4.5a shows, for comparison, traditional LIC with opacity proportional to field magnitude.

4.3.6 Interactive PFSS

Due to the absence of commonly accessible high-resolution reconstructions of the 3D coronal magnetic field, its extrapolation according to PFSS (Section 4.2.1) on the GPU is included. Because a full covering of the Sun would exceed typical time and storage constraints, this is performed on an interactive level, i.e., in interactively defined regions of interest. The researcher defines the upper left and lower right corner of the ROI with the mouse and these two points are

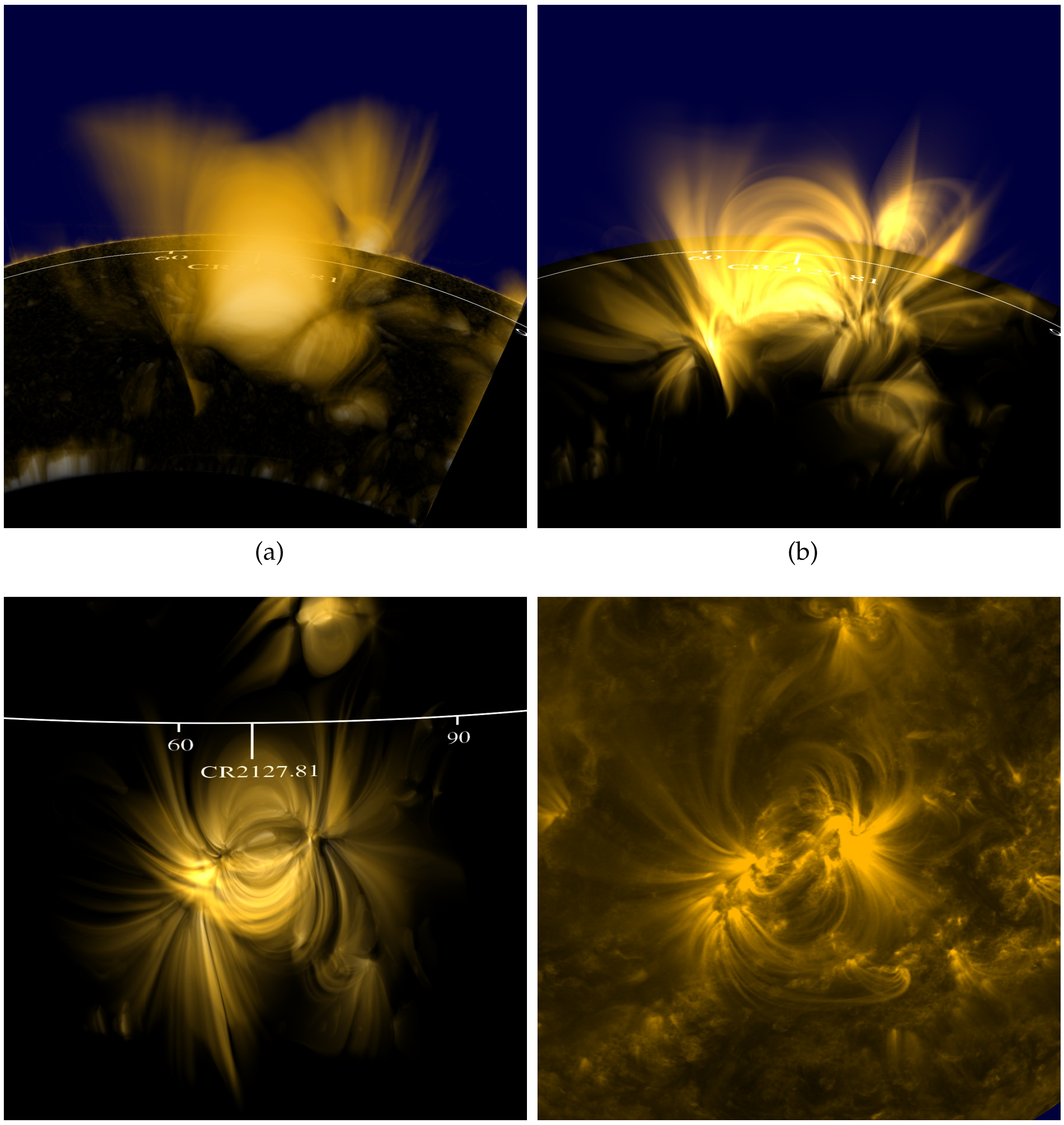


Figure 4.5 — (a) Visualizing coronal magnetic field (PFSS) with LIC using opacity proportional to field magnitude. (b) Same data visualized with virtual coronal loops. (c) Same as (b) but from SDO's view and (d) corresponding AIA 171 Å image for comparison.

transformed to LCEA coordinates on the photosphere, and the height above the solar surface is defined by numerical input. Note that typical computations take several minutes, and it is the selection of the ROI and the parametrization of

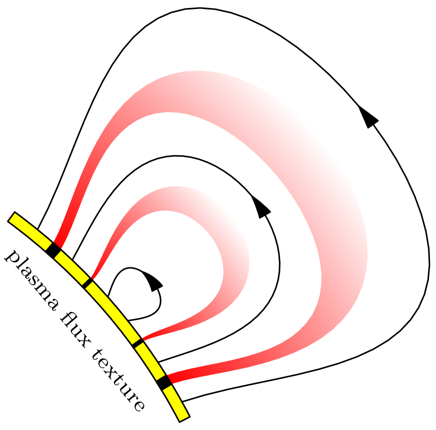


Figure 4.6 — The 'plasma flux' texture (yellow) is propagated along the magnetic field using LIC to produce the virtual coronal loops.

the reconstruction that is interactive, although quasi-interactive reconstructions rates can be achieved from small resolutions (see Table 4.1). This reconstruction serves as a fast "simulation-based visualization technique" regarding the fact that more sophisticated reconstruction techniques (e.g., NLFFF or MHD) are much more time consuming.

4.3.7 View-Dependent Visualization of Coronal Holes

Coronal holes [Cranmer, 2009] are the darkest and least active regions of the Sun, as one can see in Figure 4.7a. They are associated with rapidly expanding magnetic field structures that open up into interplanetary space and with the acceleration of the high-speed solar wind that also impacts the near-Earth space environment, see Figure 4.7b.

Interactive view-dependent visualization of coronal holes is achieved by simply combining the raytracing technique from Section 4.3.1 with the c-space field line integration from Section 4.3.2. A ray is simply shot through every pixel of the view and its nearest intersection with the solar sphere is detected. Each of these points is used as the seed of a field line, which is integrated in c-space. In contrast to Section 4.3.2, *every* point is transformed to p-space, and instead of producing a polyline from these points for display, it is simply tested if a certain distance from the photosphere has been exceeded (in the experiments a default of $1.5R_{\odot}$ according to Schatten et al. [1969] was used). If this is the case, the respective pixel is given a color (black) that encodes this behavior. If, on the other hand, a field line vertex is closer to the Sun than its second vertex, the pixel is given the color code (gray) that indicates connection to the photosphere. Despite the conservatively chosen default number of integration steps and step size, it can happen that integration does not reach one of these limits. In this case the pixel is marked with a respective color (red), indicating the user to

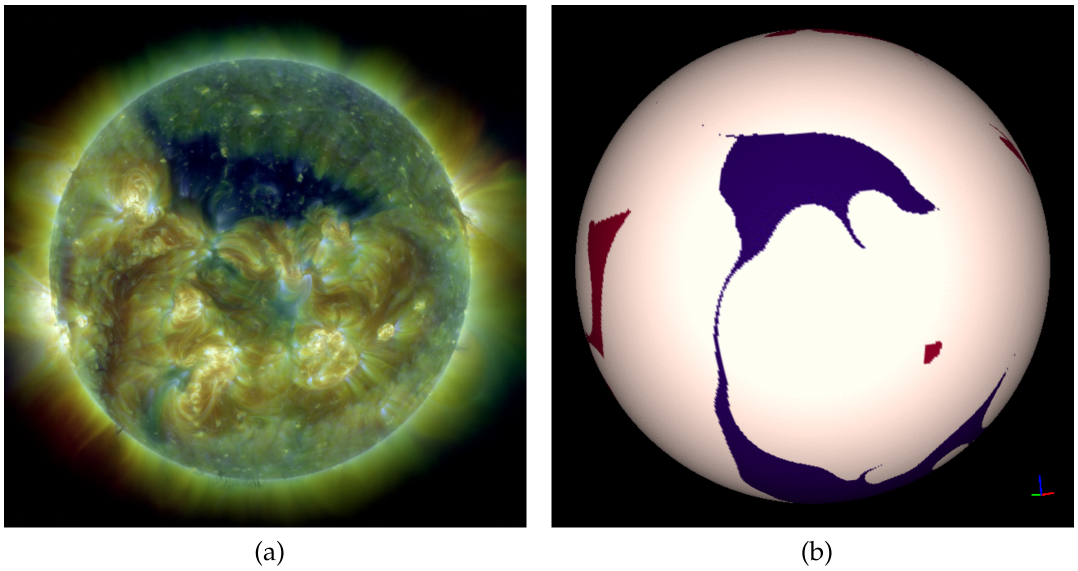


Figure 4.7 — Traditional visualizations of coronal holes at CR 2139 as dark regions at the photosphere according to X-ray images (a) and as regions at the photosphere from which the magnetic field lines reach $2.5R_\odot$ (b). (a) was generated by data from SDO of AIA 171 Å, 193 Å, and 211 Å retrieved with JHelioviewer. (b) is a courtesy of [pre].

increase the number of integration steps or increase step size.

The advantage of this view-dependent approach is that coronal holes can be inspected at arbitrary resolution at interactive rates, even with time sequences of the coronal magnetic field, which would not be possible using the precomputation in traditional approaches. See Figure 4.8 for a result (a), combined with field lines (b).

This view-dependent technique for the visualization of coronal holes, is the inspiration for the topology-based approach proposed in Chapter 5.

4.3.8 Space-Time Visualization of the Photosphere

Time-dependent solar image data are frequently displayed as time sequences or videos. Since photospheric data correspond to a narrow layer of plasma, they can be locally approximated by a 2D domain. To account for their time-dependence, they can be visualized using space-time stacking. This has, for example, been done to study so-called mesogranular flows in the photosphere using high-resolution ground-based data Roudier et al. [2003]. An efficient

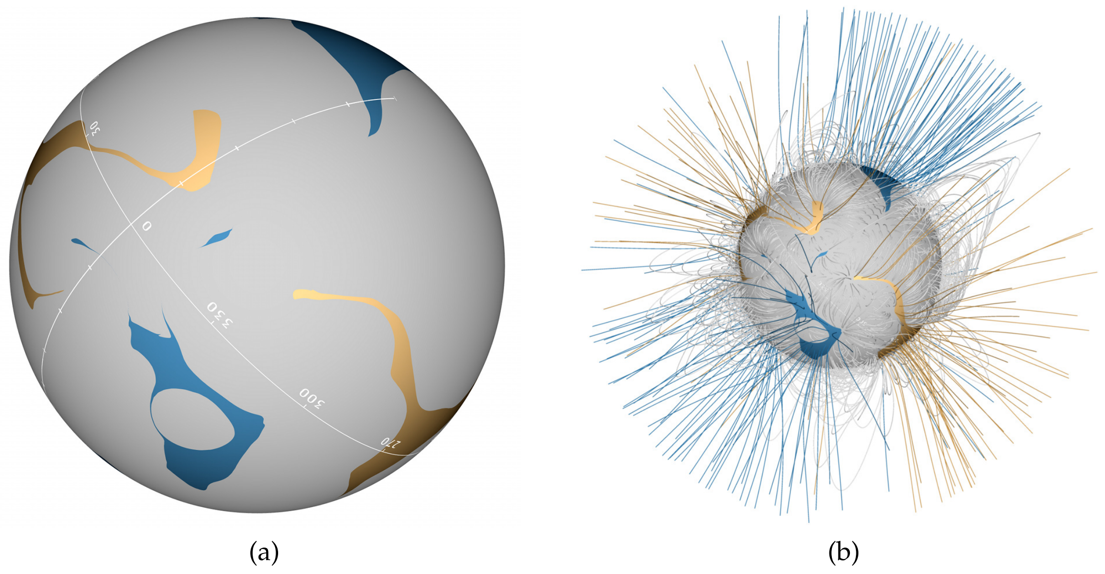


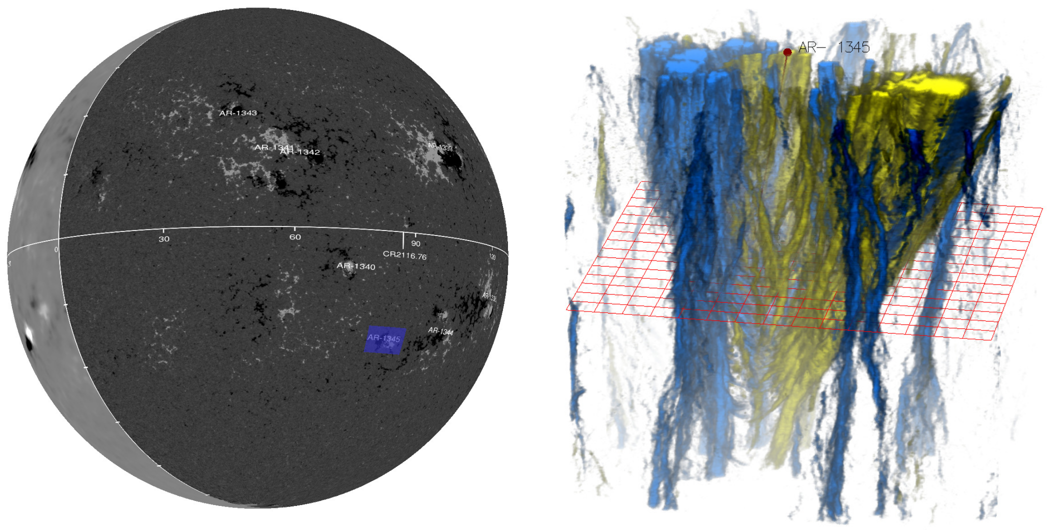
Figure 4.8 — View-dependent visualization of coronal holes. Blue/yellow regions indicate field lines reaching radius $R = 2.5R_{\odot}$ (a) in forward and reverse direction, respectively. Same as (a) with additional field lines, accordingly colored, for context (b). The magnetic field is obtained from precomputed MHD available at [pre].

approach of this functionality by volume rendering can significantly improve the ability to quickly inspect large datasets.

We stack the data in LCEA representation, using supersampling, and visualize it within a linked view using volume rendering. Furthermore, we embed a space-time view of the SRS data as polylines to provide context. Figure 4.9a shows the selected region on the solar sphere, together with the SRS information. Figure 4.9b shows the linked view containing the space-time stack of HMI magnetograms with the time instant of the solar sphere representation indicated as a red grid plane. As the user slides through time, both views are updated accordingly. By avoiding the extraction of features, such as ridge lines or isosurfaces, this approach based on volume rendering circumvents artifacts that would typically arise due to noisy data, missing data, and fluctuations.

4.4 Results

A particular focus is put on the evaluation by the domain experts. They report in Section 4.4.1 on the results from selected cases that are already used to



(a) (b)

Figure 4.9 — (a) Selection of ROI in HMI magnetogrom data. (b) Resulting space-time HMI stack (blue: south; yellow: north), over time interval of two days in spatial region of interest, actual time indicated by red grid, with additional space-time curve of active region according to SRS data (red line).

illustrate the techniques in Section 4.3. They back up their observations by astrophysics theory, evaluate the utility and possible impact of the techniques to their research area, and point out directions of future research. In Section 4.4.2, this part is concluded with a discussion of application-related results and measurements.

4.4.1 Evaluation from Domain Experts

The two domain experts evaluated the presented techniques and applications as follows.

The combination of the 3D solar spherical representation of AIA, HMI, and GONG data, together with the 3D field lines provides several advantages over existing techniques. The use of synoptic GONG data for full Carrington rotation overview, and the interactive replay of AIA and HMI data, together with their indicators in Carrington time substantially eases temporal and spatial navigation. Furthermore, browsing large datasets provides an overview of large areas of the magnetic field, which makes it possible to identify regions of interest concerning specific research questions as, for example, the connectivity between several active regions.

Although, the combination of field lines with volume rendering of magnitude is an interesting alternative, achieving appropriate field line placement for representative visualization of the coronal magnetic field is an open problem. The LIC (see Figure 4.3d), in contrast, is a promising alternative as it strongly reduces the risk of misinterpretation due to inappropriate field line placement. Nevertheless, LIC with opacity proportional to field magnitude (see Figure 4.3c) seems to have particularly high potential in solar research because it shares the advantages of LIC but reduces clutter. Promising further work could apply LIC to more complex solar magnetic field models, which can contribute to the study of topological features like magnetic separatrices and null points that are thought to play a key role in coronal dynamics.

Table 4.1 — Performance measurements and GPU memory consumption in MiB for PFSS calculation.

Resolution $(\varphi, \sin\theta, r)$	GPU Mem	Computation time
$50 \times 50 \times 20$	75 MiB	4.8 s
$50 \times 50 \times 60$	94 MiB	26 s
$100 \times 100 \times 60$	183 MiB	1 min 13 s
$150 \times 150 \times 60$	332 MiB	2 min 4 s
$200 \times 200 \times 60$	539 MiB	6 min 1 s
$250 \times 250 \times 60$	808 MiB	9 min 22 s

The technique of virtual coronal loops represents a very interesting approach for the research of coronal plasma dynamics. Observational studies of magnetograms have shown permanent changes in the photospheric magnetic field after flares. 3D visualizations of the before/after stage of such "collapsing" magnetic loop systems and especially the comparison of virtual coronal loops with observational data would provide a big step in validating the model and obtaining high number statistics. The fully interactive exploration is likely to inspire new research questions and physical models and future work shall investigate new possibilities of coronal plasma reconstruction and interactive visualization.

The integrated PFSS provides a convenient approach for extrapolating the 3D coronal field. The fact that the results can easily be visualized together with, e.g., GONG SH data allows for interactive exploration based on PFSS.

The solar wind carries magnetic flux from the photosphere into the heliosphere originating from topologically open areas. While open magnetic flux is unevenly distributed at the solar surface, its distribution is much more uniform in the outer corona, at least during periods of low solar activity. Standard potential field models do not provide such uniform distribution of open flux in the heliosphere and alternative approaches have been suggested, e.g. by Gilbert et al. [2007] and Wang and Sheeley [1992], point out that the observed photospheric field should first be corrected for line-of-sight projection and then matched to the radial component of the potential field. In general, there appears to be a firm causal link between the largest coronal holes and high-speed solar wind streams [Cranmer, 2009], but especially the role of small-scale magnetically open regions and their linkage to the heliosphere is not fully understood. The fact that the view-dependent visualization technique presented here works at interactive frame rates at full screen resolution, allows for detailed inspection of areas of open magnetic field at different spatial scales and for interactive zooming over several orders of magnitude. This makes it very attractive for further studies of the topological properties of the Sun's large-scale magnetic field and their relation to the different types of solar wind that vary significantly in speed and composition. And it provides a powerful tool for research of coronal holes.

The space-time visualization technique has proven its potential as compared to traditional replay of AIA and HMI image sequences. Having the space-time structure at instantaneous view, together with the active regions from SRS, provides new and exciting insight in the dynamics of the convection-dominated magnetic field, in particular in the vicinity of sunspots. Specific research questions, such as the behavior of magnetic elements that persistently stream away from sunspots, so-called moving magnetic features (MMF) require the analysis and inspection of large areas in magnetograms and their temporal evolution. The simultaneous spatiotemporal visualization provides the means to identify patterns in the behavior of MMFs which are difficult to recognize in video displays.

4.4.2 Discussion

For the coronal magnetic field extrapolation on the GPU with PFSS, we use the Cusp library [Bell and Garland, 2012], which is still in a development state and which lacks suitable preconditioners. This makes the convergence of our PFSS code rather slow. More advanced libraries, like the GPU Accelerated Linear Algebra (CULA) library that is implemented in CUDA, are likely to reduce the computation time. Moreover, for high resolution PFSS and LIC calculations, the GPU memory consumption reaches the limits of todays graphics hardware.

For the performance measurements of this application, a Linux based system, which consists of an Intel(R) Core(TM) i7-2600 CPU @ 3.4 GHz, an NVidia

Table 4.2 — Performance measurements for the precalculation of the LIC volume.

Method	Resolution	time(sec)	# steps
LIC (Fig. 4.3(d))	$900 \times 450 \times 250$	36.18	250
LIC (Fig. 4.3(d))	$360 \times 180 \times 100$	2.70	250

Geforce GTX 560 Ti graphics card with 2 GiB RAM with NVidia driver version 295.20, and CUDA version 4.1 is used. Table 4.1 shows the performance and GPU memory consumption for the PFSS calculation, where we have used the following region of interest: 2011-01-22, 06:00:00Z, $r/R_\odot = [1, 1.25]$, $\sin\theta = [-0.05, 0.8389]$, $\varphi = [50°, 80°]$. Table 4.2 shows the computation times for generating the LIC volumes. The example of volume rendering (Figure 4.3b), with a resolution of 1200×1200 and 500 steps per ray performs with 2.48 fps. The view-dependent approach for the visualization of coronal holes runs at 1.44 fps in a screen resolution of 900×900. Please note that no acceleration technique has been used.

CHAPTER 5

Escape Maps

Our concept of escape maps is inspired by the definition of coronal holes as presented in Chapter 4. Here, they are formulated as follows: given a bounded vector field (or a subregion of an unbounded vector field), the boundaries are divided into *map boundary* and *escape boundary*. The *escape map* is then defined by a spatial representation at the map boundary with respect to the connectivity with the escape boundary by means of streamlines. In other words, it provides a map on the map boundary indicating if a streamline seeded at the respective position reaches the escape boundary or not, or conceptually whether they escape from the domain through the escape boundary. Thus, escape maps do not map to the position of escape but they provide a map *if* a respective streamline escapes from the domain. In this chapter, those parts of the map that escape are referred to as *escape regions* and those that do not escape as *rest regions*. The boundaries between escape regions and rest regions are denoted *escape region boundaries*. Applications of escape maps include, for example, the investigation of the outreach of a fluid flow or cases where one is interested if a substance can leave a region under the transport of a flow, like in the context of pollution or supply.

A straightforward implementation to obtain the streamline-based mapping between the map boundary and the escape boundary would be to densely seed streamlines on the map boundary, test if they reach the escape boundary in forward or reverse direction of integration, and label the seed points accordingly. This approach has been already used in the astrophysics community to analyze *coronal holes* (see Chapter 4), which represent regions of the Sun's

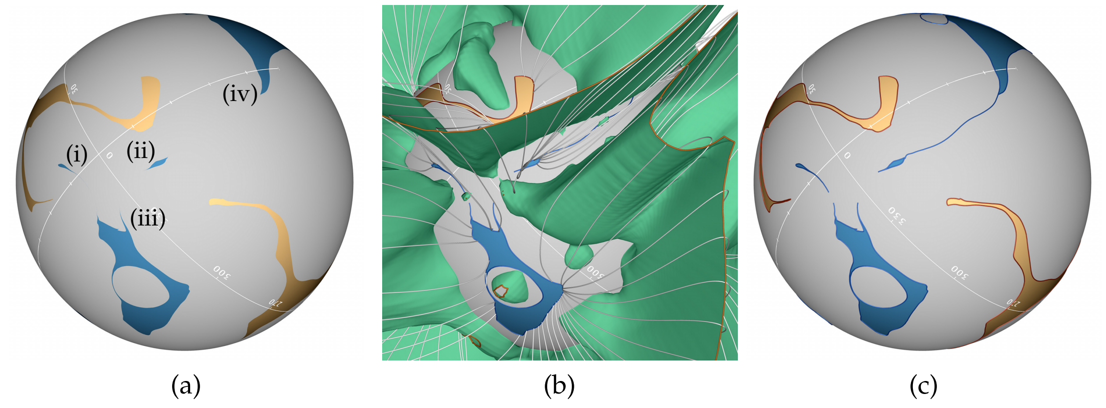


Figure 5.1 — Escape maps at the example of coronal hole extraction. (a) View-dependent visualization of coronal holes (yellow, light blue) by directly seeding field lines at the photosphere (Section 4.3.7) suffers from discretization problems. The visualization result suggests that (i), (ii), and (iii) might be connected by corridors, but it is impossible to resolve them with this approach. (b) The escape map extraction obtains coronal hole boundaries (red, blue) by seeding field lines at boundary curves (orange) of trimmed isocline surfaces (green). (c) Isocline-based escape map extraction captures the full structure of coronal holes, revealing the corridor between (ii) and (iv).

photosphere with reduced ultraviolet or X-ray emission, and are defined in this context as those regions on the photosphere that exhibit streamlines of the solar magnetic field that reach, in forward or reverse direction, an escape radius which is larger than the radius of the photosphere. In this configuration, the photosphere represents the map boundary, and the escape radius defines the escape boundary. The extraction of coronal holes is one of the applications of this approach that is demonstrated in this chapter, among examples from computational fluid dynamics and magma convection in the Earth's mantle. The major difficulty with this direct sampling approach (see Figure 5.1a) is, however, that it is typically not able to reveal the correct structure of the escape map because this map typically exhibits extremely thin features which cannot be appropriately sampled with a regular sampling (see Figures 5.1 and 5.2). At the same time, these fine structures typically exhibit diverging streamlines, which leads to strong error accumulation during integration from the map boundary and hence inaccurate or even useless streamlines and resulting mappings.

Vector field topology represents the transport in vector fields by separating regions of coherent streamlines' behavior, and the traditional approaches, as mentioned in Section 2.5, depends on critical points and periodic orbits for complete representation with their manifolds. The extraction of escape maps consists of a problem directly related to vector field topology, because they define regions of coherent streamlines' behavior as well (i.e., regions with streamlines that escape from the domain or not), and the separatrices in this context represent escape region boundaries. However, the traditional vector field topology is only sufficient in unbounded domains, as it demands that the involved critical points and periodic orbits to be available (i.e., located inside the domain) for completeness. Moreover, as shown in Section 5.1.4, existing approaches that address bounded vector field topology, i.e., boundary switches (Section 2.5.4) and bifurcation lines (Chapter 3), are not sufficient to extract escape maps and are likely to produce false positives.

In this chapter, an approach that is intrinsically independent of constructs such as critical points or periodic orbits is presented. It can determine streamline-based connectivity between the boundary of arbitrary simply-connected regions Ω of the domain, and hence extends traditional vector field topology. To this end, the map boundary where streamlines are conceptually seeded, is defined by one simply-connected part of the boundary of Ω, and the remaining part of the boundary is the escape boundary. This technique then provides a topologically correct escape map by consistently extracting all escape region boundaries.

Furthermore, this chapter shows the close connection between the topological concept of escape maps and isocline surfaces of the vector field, and derive from this connection a sampling technique for escape maps that guarantees their topologically correct extraction. Since this approach extracts only escape region boundaries, i.e., the boundary curves between escape regions and rest regions of the escape map, as one can see in Figure 5.1c, this technique is complemented with a view-dependent implementation of the straightforward sampling approach, demonstrated in Figure 5.1a, which is similar to the view-dependent technique for the visualization of coronal holes, as presented in Section 4.3.7.

Specifically, the contributions of this chapter include:

- The introduction of escape maps to scientific visualization,
- a topologically correct and robust extraction technique for the escape region boundaries within escape maps,
- derivation of the connection between isocline surfaces and concepts from vector field topology, and
- a view-dependent implementation for direct sampling-based visualization of escape maps.

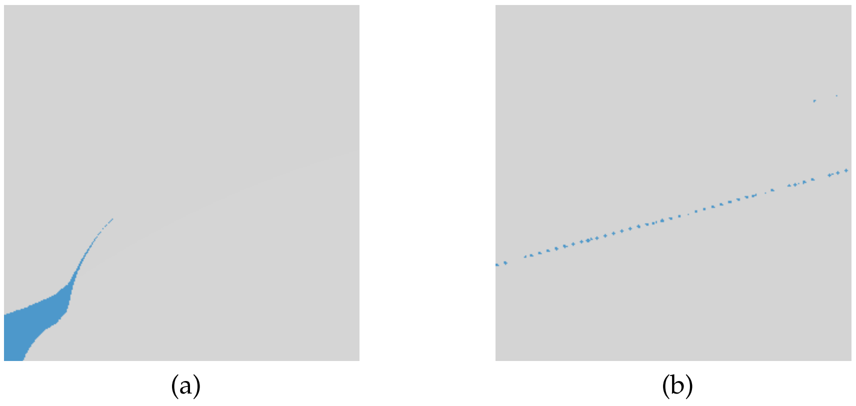


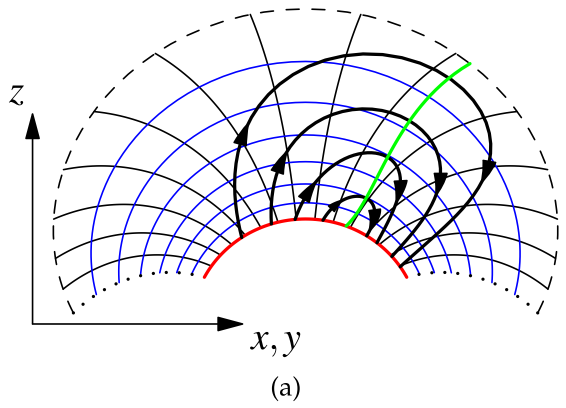
Figure 5.2 — (a) Closeup of Fig. 5.1a at right-upper tip of coronal hole (ii), at zoom factor 16. It is impossible to judge whether there is a corridor between (ii) and (iv). (b) View at the center of (a) at zoom factor 5,000. The corridor appears here because its width is wider in vicinity to (ii). However, numerical precision issues are apparent, prohibiting further zooming and hence identification of the corridor farther away from (ii).

5.1 Extraction of Escape Region Boundaries

The following terms and symbols are used specifically along this chapter. The *map boundary* and *escape boundary* are denoted $\partial\Omega_M$ and $\partial\Omega_E$, respectively. Both have to be simply connected and together they enclose a simply-connected region Ω. If a given streamline seeded on $\partial\Omega_M$ reaches $\partial\Omega_E$ in forward or in reverse direction, it is referred to as *escape streamline*, otherwise as *rest streamline*. Note that, rest streamlines can also exhibit infinite trajectories within the domain and reach neither of the boundaries.

As a prerequisite for the formulation that will follow, Ω needs to be provided with a regular parametrization $\mathbf{x}(\xi,\eta,\zeta)$. (Here and in the following, boldface symbols are used to indicate vector-valued functions.) In this parametrization, the map boundary $\partial\Omega_M$ is represented by the set $\{\mathbf{x}(\xi,\eta,0)\}$ and the escape boundary $\partial\Omega_E$ is composed of the subsets $\partial\Omega_t = \{\mathbf{x}(\xi,\eta,1)\}$ and $\partial\Omega_s = \{\mathbf{x}(\pm 1,\pm 1,\zeta)\}$ if Ω is not periodic in ξ and η. Thus, $\mathbf{x}(\xi,\eta,\zeta)$ maps the computational space (c-space) coordinates ξ, η, ζ to the physical space (p-space) Cartesian coordinates x, y, and z. Figure 5.3 illustrates one example of such transformation, where (b) is the p-space representation of (a) in p-space.

If Ω consists of a rectangular domain, determining a regular parametrization $\mathbf{x}(\xi,\eta,\zeta)$ is straightforward. However, if Ω has generic simply-connected shape, the problem is equivalent to meshing Ω with a structured grid. Follow-



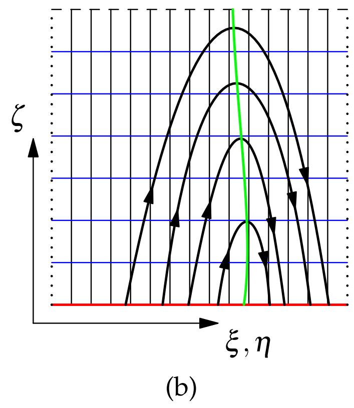


Figure 5.3 — Parametrization $\mathbf{x}(\xi, \eta, \zeta)$ of the simulation domain Ω with illustrated streamlines (black) and corresponding isocline surface (green). (a) Arbitrary simply-connected region in physical space with map boundary $\partial\Omega_M$ (red) and escape boundary $\partial\Omega_E$ (dashed). The dotted lines can serve either as escape or as periodic boundaries. (b) Computational space represented by the parameters (ξ, η, ζ).

ing Thompson et al. [1998], this could be accomplished by solving a boundary-value problem of the Laplacian equation, for example. A detailed discussion of mesh generation is, however, beyond the scope of this work.

The approach presented in this chapter extracts escape region boundaries, i.e., the curves on $\partial\Omega_M$ that separate escape regions from rest regions. In the first step of this technique, the escape region boundary curves are extracted based on isocline surfaces of the vector field (Section 5.1.2). This step typically extracts the majority of the boundary curves. The remaining curves are extracted based on vector field topology features within Ω (Section 5.1.3). Algorithmic (discretization) details are given in Section 5.2.

5.1.1 Motivation

The straightforward approach to visualize escape maps is by *direct sampling*. This approach consists of seeding a set of streamlines on $\partial\Omega_M$ and color their starting point with one of three different colors: (i) indicating that the streamline reaches $\partial\Omega_E$ in forward direction, (ii) indicating that the streamline reaches $\partial\Omega_E$ in reverse direction, and (iii) indicating that the streamline does not reach $\partial\Omega_E$. Here, blue, yellow, and gray are used to represent (i), (ii), and (iii), respectively. Note that if the streamline reaches $\partial\Omega_E$ in both directions, it represents an inbound boundary switch curve [Weinkauf et al., 2004], which is unlikely to

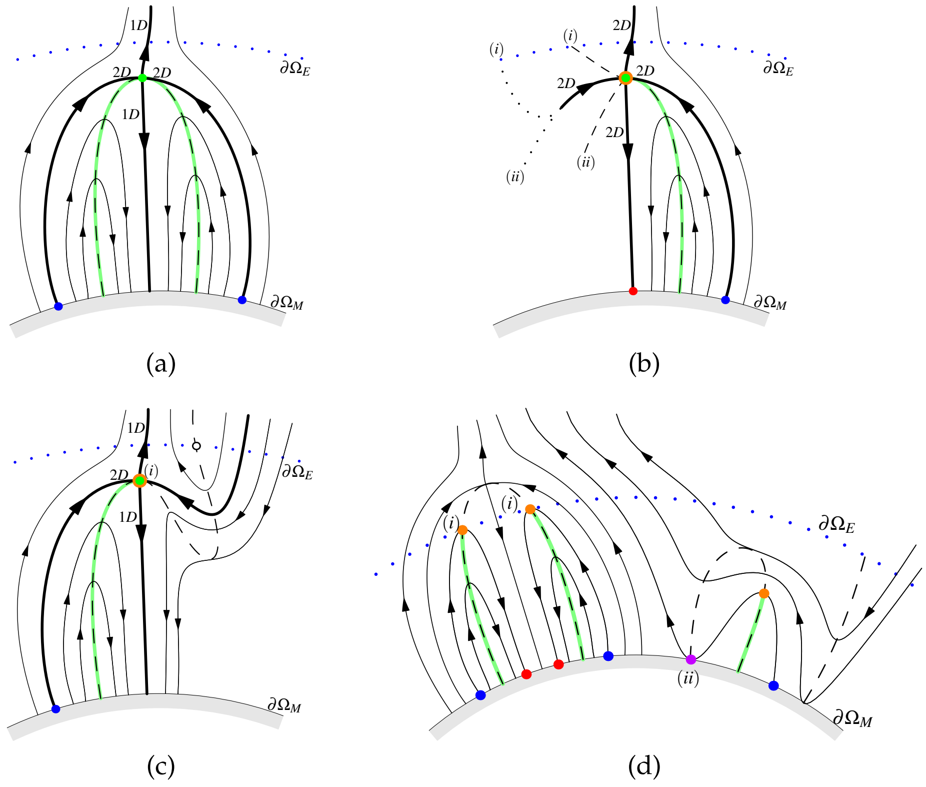


Figure 5.4 — Cross-sectional illustration of different 3D cases. If topological features (e.g., saddle points or saddle orbits, green dots) reside within Ω, then escape region boundaries (red, blue) are obtained by intersecting 2D manifolds of saddle-type critical points (a) or periodic orbits (b) with the map boundary $\partial\Omega_M$ (light gray). (d) Otherwise, escape region boundaries are obtained by intersecting $\partial\Omega_M$ with streamlines starting at boundaries (orange) of trimmed isocline surfaces (green curves). If this boundary resides at $\partial\Omega_M$ (purple point (ii) in (d)) there may be only one streamline connecting from (ii) to $\partial\Omega_E$. This boundary (purple) also represents a part of the escape region boundary and an inbound boundary switch curve. (c) If a critical point resides within Ω but part of its 2D manifold reaches the escape boundary $\partial\Omega_E$, there is also an isocline surface reaching $\partial\Omega_E$ and the 'top' (orange) of its trimmed part generates the escape region boundary.

be consistently sampled with the direct approach. However, since such curves do not separate escape regions from rest regions (in fact they separate forward escape regions from reverse escape regions), they are of no particular interest. Nevertheless, the extraction technique presented in this chapter can obtain those curves as well. The straightforward sampling approach is traditionally applied in a static manner, i.e., a texture is built with these colors and mapped to the physical space. Unfortunately, this approach cannot extract very narrow escape regions (called corridors), as illustrated by Figures 5.1a and 5.2. Note that, in favor of the direct sampling approach, this chapter compares the results of the proposed technique to a view-dependent implementation of the direct sampling (similar to the technique for coronal holes presented in Section 4.3.7), i.e., the figures in this chapter show a 'texture' that is computed interactively at screen resolution. This allows the user to zoom at the desired features interactively. There are two major reasons why the straightforward approach still fails at resolving the corridors: the corridors are extremely narrow and hence extremely high zoom factors would be required, which would lead to numerical issues with sampling, and secondly, streamline integration is subject to numerical issues too (see Figure 5.2), even with double-precision arithmetic. This is of particular impact in these configurations as the growth of integration error is much higher in the direction from corridors on $\partial\Omega_M$ compared to integration toward corridors on $\partial\Omega_M$, because streamlines have to diverge in direction away from $\partial\Omega_M$ to cause a narrow escape region, and this diverging streamline behavior represents unpredictable advection and hence typically impedes proper integration. Therefore, it is very difficult to discretize the fine escape regions structures on $\partial\Omega_M$ in a pixel-based manner. In contrast, it turns out to be much simpler to generate seeding constructs at some distance from $\partial\Omega_M$ and to obtain therefrom a polyline representation of the escape region boundaries, because streamlines converge toward the corridors in this direction of integration and thus are typically highly accurate.

5.1.2 Isocline Surface-Based Extraction Step

Escape maps are defined by the topological connectivity between $\partial\Omega_M$ and $\partial\Omega_E$. Since topological properties are invariant under continuous deformation, the topology in c-space and p-space are dual, i.e., the streamline connectivity between $\partial\Omega_M$ and $\partial\Omega_E$ is equivalent in both physical and computational space. Hence, this extraction technique can operate in computational space and generate consistent result in physical space. Throughout this chapter, unless explicitly stated, the velocity field in computational space is referred to as 'vector field' only. In the following, the utility of isocline surfaces is discussed and subsequently the extraction technique is presented followed by its discussion.

Motivation

Isocline surfaces consist of the loci where a vector field exhibits a predefined slope. Zero-slope isocline surfaces can be obtained by computing the dot product between the vector field and the 'up' vector (in this case $(0,0,1)^\top$ in computational space) and extracting the zero-level isosurface from the resulting scalar field, denoted *isocline surface* or *isocline* henceforth. See the dashed/green curves in Figures 5.3 and 5.4 for illustrations of zero-slope isoclines.

Since the rest streamlines, i.e., those that do not escape, are continuously differentiable and bounded by $\partial\Omega$, they must (i) converge to a critical point, (ii) find some kind of infinite path inside Ω, e.g., toward a periodic orbit, or (iii) rise from $\partial\Omega_M$ and return to $\partial\Omega_M$ again. Since critical points are zeros of the vector field, they must reside on isocline surfaces. The case (ii) of an infinite path in a bounded region implies a return of the streamline, and, thus, in the non-degenerate case, a zero slope in at least two and at most an even number of points along the curve. The case (iii) must, due to the mean value theorem, exhibit an odd number of zero-slope points along the streamline. Hence, all cases exhibit at least one intersection with the isocline surface. Therefore, any rest streamline seeded on $\partial\Omega_M$ will intersect the isocline surface at least once. If one rejects all parts of isocline surfaces that are intersected by an escape streamline, a *trimmed isocline surface* Θ is obtained, and its boundary curves $\partial\Theta$ are necessary and sufficient seeding structures to obtain all rest streamlines that connect to an escape region boundary. Once $\partial\Theta$ are propagated along the vector field both in forward and reverse direction, their intersection with $\partial\Omega_M$ will be at an escape region boundary. Note that boundary curves $\partial\Theta$ that are directly located on $\partial\Omega_M$ are either between escape and rest regions or between forward and backward escape regions (and in both cases inbound boundary switch curves).

Technique

1. Extract all isocline surfaces with slope zero relative to $\partial\Omega_M$, i.e., extract the zero-level isosurface of the 'up' component of the vector field within Ω. Note that the isosurfaces are closed or exhibit a boundary at $\partial\Omega_E$ and/or $\partial\Omega_M$.

2. Erode all parts of these surfaces that exhibit escape streamlines.

3. For each remaining surface part, extract its boundary curve.

4. Use these boundary curves as seeds for streamsurfaces.

5. The intersection curves of all streamsurfaces with $\partial\Omega_M$ represent escape region boundary parts.

Figure 5.1c shows the final result after merging the (partial) escape region boundaries obtained in this step with the results from the topology-based step (Section 5.1.3).

Discussion

The approach described in Section 5.1.2 cannot produce false positives in a continuous formulation. The cases depicted in Figures 5.4 (a)–(c) are simplified (degenerate)—in general, a critical point, periodic orbit, or bifurcation line would never be located exactly at the 'highest' point of an isocline surface with its 2D manifold aligned with the isocline. Thus, in general, at least a small part of the isocline surface is located above the 2D manifold and hence eroded, leaving a 'hole'. Integrating a streamsurface from the boundary of this hole in forward direction converges to the critical point, periodic orbit, or bifurcation line, since in the continuous case, this boundary coincides with the 2D manifold. However, in a discretized formulation (the implementation), the isocline surface is eroded a bit too much due to discretization and hence the stream surface is likely to miss the critical point, periodic orbit, or bifurcation line. It will rather deviate to the bottom and intersect $\partial\Omega_M$ at the red point in Figure 5.4b, leading to a false positive. The reverse streamsurface, on the other hand, will successfully extract the blue points in Figures 5.4 (a)–(c). Therefore, in the implementation, false positives are avoided by integrating streamsurfaces seeded at isocline boundaries only in direction opposite to the streamlines direction that caused the erosion of that isocline boundary (no streamsurface is seeded in case of forward and reverse escape). Nevertheless, to assure robust extraction of the escape region boundaries, the topology-based approach, as presented in Section 5.1.3, is applied in the discretized implementation to complement the isocline-based approach.

5.1.3 Topology-Based Extraction Step

Because escape regions give rise to streamlines that reach $\partial\Omega_E$, in contrast to those seeded at rest regions, these two regions exhibit qualitatively different streamline behavior. Hence, extraction by concepts from vector field topology is an obvious choice. One such concept are separatrices, i.e., manifolds of streamlines converging in forward or reverse direction to saddle-type critical points or saddle-type periodic orbits (Section 2.5). They represent barriers with respect to qualitatively different streamline behavior and should therefore be able to provide the boundaries of escape regions. Note that a saddle-type critical point, i.e., an isolated zero of the vector field with opposite signs of the real eigenvalue parts of the Jacobian, exhibits one 2D manifold and one 1D manifold, while a saddle-type periodic orbit, i.e., an isolated closed streamline with saddle-type behavior in its mapping under a full revolution, exhibits two 2D manifolds. Note that, for simplicity, in this chapter the twisted-saddle periodic orbits are referred to as saddle-type periodic orbits.

In an unconstrained setup ($\partial\Omega_E \to \infty$), one could define escape streamlines as those that reach infinity. Escape region boundaries could then be obtained by intersecting the 2D separatrices of saddle points and saddle orbits with $\partial\Omega_M$ (cross-section illustration in Figures 5.4 (a) and (b)), with the additional requirement that, in case of a saddle, one end of its 1D manifold connects to $\partial\Omega_M$ and the other reaches $\partial\Omega_E$, and that in case of a saddle-type periodic orbit, both 2D manifolds connect to $\partial\Omega_M$ while at least one of its 2D manifolds reaches $\partial\Omega_E$. This approach, to some extent, is also possible in constrained setups (with non-empty $\partial\Omega_E$) as long as the required critical points and periodic orbits reside within Ω. Hence, this approach is used for all saddle-type critical points within Ω. However, as discussed above, this approach alone (without the isocline-based approach) would be insufficient even in the unconstrained setup because boundary switch curves (Figures 5.4d (ii)) can also give rise to non-escape streamlines in absence of a saddle-type critical point, and bifurcation lines too. Furthermore, if a 2D manifold intersects $\partial\Omega_M$ only in one area but reaches $\partial\Omega_E$ in an other area, as illustrated in Figure 5.4c, there is always an isocline surface also reaching $\partial\Omega_E$ and hence the respective escape region boundary (part) is already extracted in Section 5.1.2.

Technique

1. Extract all saddle-type critical points within Ω.
2. For each saddle, one end of its 1D manifold has to connect to $\partial\Omega_M$ and the other has to connect to $\partial\Omega_E$ (Figure 5.4a), otherwise the saddle is rejected.
3. Finally, intersect all 2D manifolds of the remaining saddles with $\partial\Omega_M$. The resulting intersection curves represent the missing escape region boundary parts.

Figures 5.5a and 5.12b give an example of the (partial) escape region boundaries obtained in this step.

Discussion

As discussed in Section 5.1.2, the isocline-based approach may miss 2D manifolds of critical points due to discretization issues and are therefore extracted explicitly in Section 5.1.3. The main difficulty with isocline-based extraction

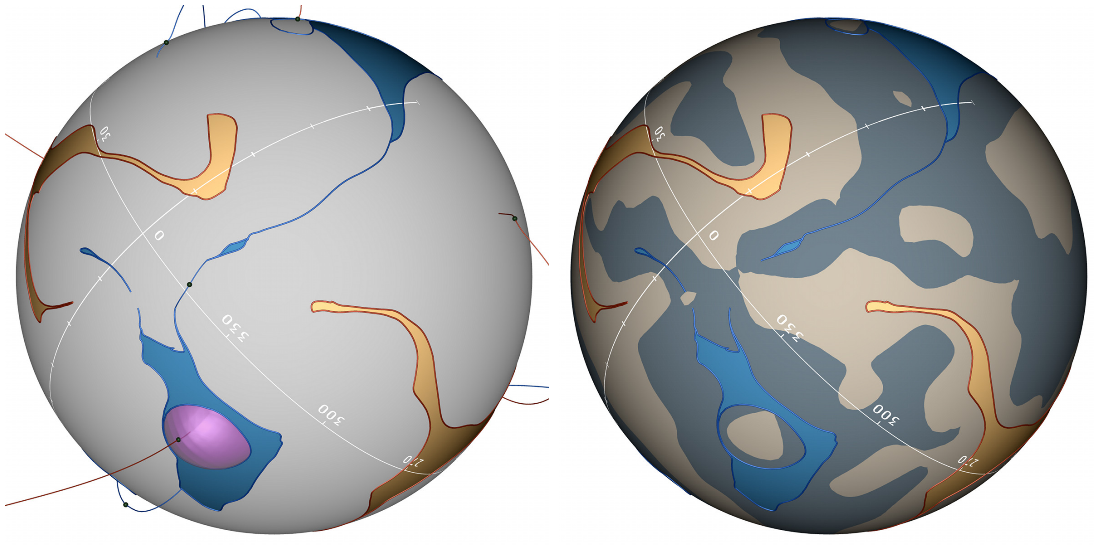


Figure 5.5 — MAS data at Carrington rotation 2128. (a) Same as Figure 5.1, but with critical points (green) and their 1D (blue stable, red unstable) and 2D (transparent pink) manifolds. It is apparent that the topology-based step provides only very few boundaries (i.e., the intersection of the 2D manifold with the photosphere $\partial\Omega_M$) of the coronal holes. (b) Same as Figure 5.1c but with polarity of the photosphere, i.e., radial magnetic flux B_r at photosphere, visualized. One can see that the three blue coronal holes belong to the same connected component of 'north' polarity.

of 2D manifolds of saddle-type critical points is that the critical points are always located on the isocline surfaces and that, due to discretization issues, the erosion may fail to erode those parts of the isocline surfaces that contain the critical points. The circumstances are, however, different for saddle-type periodic orbits. A non-degenerate (i.e., not perfectly 'horizontal') periodic orbit intersects isocline surfaces only at isolated points. Hence, for reasons analogous to those discussed above with Figure 5.4c, saddle-type periodic orbits are robustly extracted already with the isocline-based approach (see also the bifurcation line-based evaluation in Section 5.1.4).

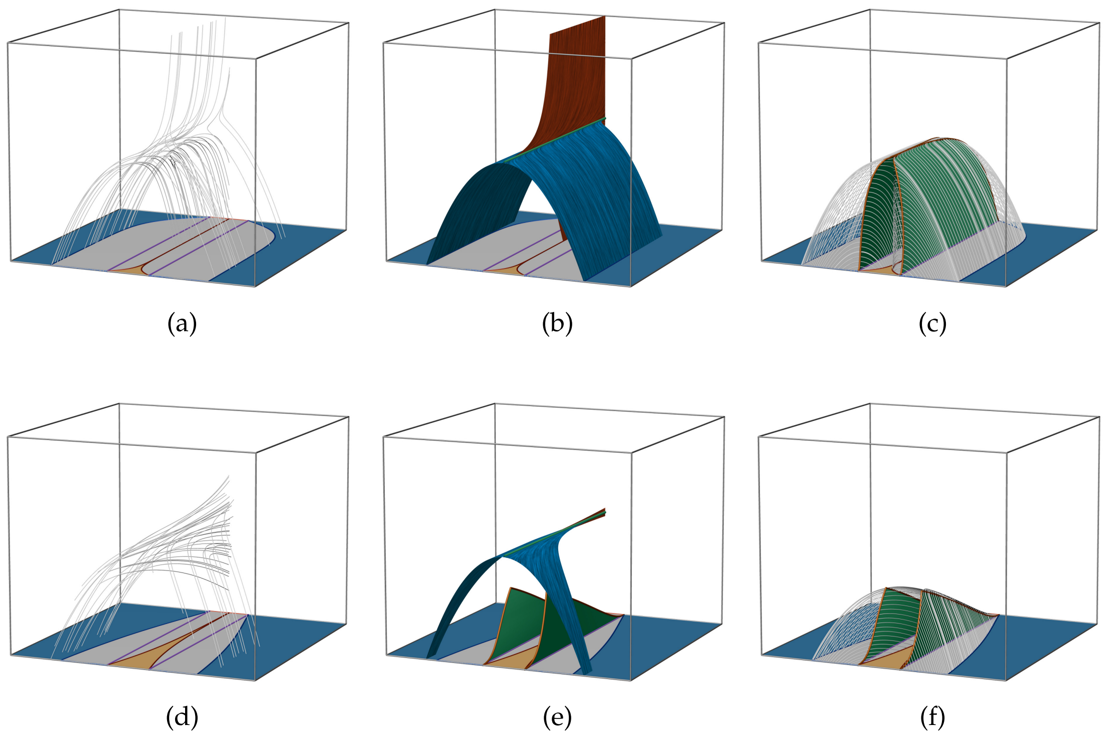


Figure 5.6 — Analytic field, Equation 5.1, with $\lambda = 2$ ((a)–(c)) and $\lambda = 10$ ((d)–(f)). (a) Some streamlines together with escape map from our approach. Boundary switch curves [Weinkauf et al., 2004] (violet) are all outbound and not consistent with escape map, hence not able to extract escape map. (b) Bifurcation line [Machado et al., 2013] (green) exhibits stable (blue) and unstable (red) manifold, which intersect map boundary at escape region boundaries. However, they provide only small straight parts. (c) Our eroded isocline surface (green) with streamsurfaces (illustrated by selected streamlines) seeded at its boundary provides the complete map. (d)–(f) Corresponding case for $\lambda = 10$. Note that bifurcation manifolds fail here too (also because isoclines do not reach bifurcation line) but our approach succeeds.

5.1.4 Evaluation

First, a short comparison to previous work is provided, i.e., to boundary switch curves and their separation surfaces [Weinkauf et al., 2004] on the one hand, and bifurcation lines and their manifolds [Machado et al., 2013] on the other hand. The comparison is based on the following vector field:

$$\mathbf{u}(\mathbf{x}) := \left(-x, 4 - (0.4x)^2 - (0.1y)^2, \lambda\right)^\top . \tag{5.1}$$

As illustrated in Figure 5.6, this field does neither contain a critical point nor a periodic orbit, but it exhibits a bifurcation line (green line) in z-direction. Its stable manifold (blue) intersects the y-max plane with its both sides, and in the case where $\lambda = 2$ the unstable manifold (red) intersects both the y-min and y-max plane of the shown region $[-30, 30] \times [-45, 5] \times [-30, 30]$. For the tests, the region is discretized on a Cartesian grid with resolution of $60 \times 50 \times 60$ nodes.

It can be seen that the escape map (which is located at $y = 5$) of this field exhibits a corridor, and that neither the outbound boundary switch curves (and their separation surfaces) nor bifurcation manifolds can extract the escape map here, and hence in general configurations. The isocline-based approach is the only technique that robustly extracts escape maps.

This evaluation is concluded by examining the structural stability of escape maps, i.e., their robustness against perturbations. At the same time this gives a notion on the numerical stability of our extraction approach. To this end, different levels of white noise is added to Equation 5.1 with $\lambda = 2$ (see Figure 5.8). While again, the boundary switch curve approach fails in both cases, the bifurcation manifold approach is not substantially affected with 2% noise. However, with 30% noise, the bifurcation line or its manifolds could not be extracted. Still, this isocline-based technique extracts an escape map without disruptions or other artifacts and the streamlines are consistent. Nevertheless, such an analysis can only give an impression of the structural stability versus numerical stability. Deriving the structural stability of bifurcation lines and relating them to integration error would exceed the scope of this work. Taking into account persistence could be also a promising direction of research. Furthermore, manifolds from boundary switches at escape boundary when reaching the map boundary are likely to produce false positives, if the other side of this manifold returns to the escape boundary, as illustrated in Figure 5.7.

5.2 Implementation Details

The description of the technique was, up to now, mostly in terms of continuous concepts. Here, implementation details are provided, in particular regarding discretization. The entire implementation works in double precision, including the parts on the GPU.

5.2.1 Parameters

While the approach of direct sampling-based escape map extraction provides one conceptual parameter, the escape boundary $\partial\Omega_E$, the implementation of this

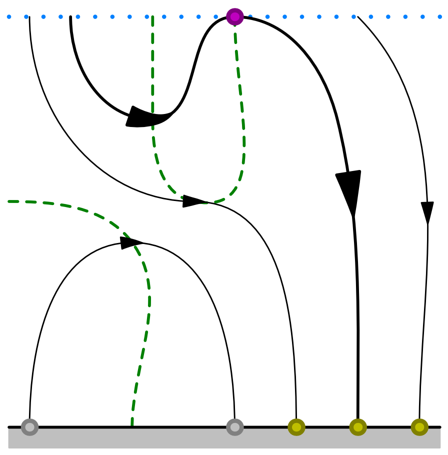


Figure 5.7 — Cross sectional representation of a vector field, where a boundary switch (purple) at the escape boundary (dotted blue at top) produces a false positive at the map boundary (black at bottom). Isocline surfaces (green) are illustrated for context.

approach requires a few additional parameters. The primary parameter is the discretization size of the resulting escape region boundaries. It is controlled by the maximum length δ of the segments of the resulting polyline representation. This parameter is used for inserting new trajectories during streamsurface computation (Section 5.2.3). Further parameters are the step size for line integration, the cell and mesh subdivision levels for isocline extraction, the maximum number of bisection steps for trimming (erosion), and the number of subdivision steps for the extraction of critical points.

5.2.2 Data Representation

Because the prototype operates almost entirely on the GPU and to ensure maximum performance, it supports in its present state any type of structured grid, i.e., any curvilinear grid. Nevertheless, extending the implementation for data on unstructured grids is straightforward.

The Buoyant Flow example covered in Section 5.3.1 is given on a uniform grid, while the Solar MHD data presented in Section 5.3.2 are given on spherical curvilinear grids, and the Magma Flow dataset, covered in Section 5.3.3, is also given on a spherical curvilinear grid.

5.2.3 Streamsurface Integration

Although the complete streamsurface is provided for illustration purposes of the 2D manifolds, this approach does not require an explicit surface representation. The ordered set of streamlines is sufficient to obtain the intersection curves between the streamsurfaces and $\partial\Omega_M$. This allows to follow the rather simple approach due to Hultquist [1992], but it also allows to omit the triangle generation step, similar to the technique presented in Chapter 6. Hence, it is

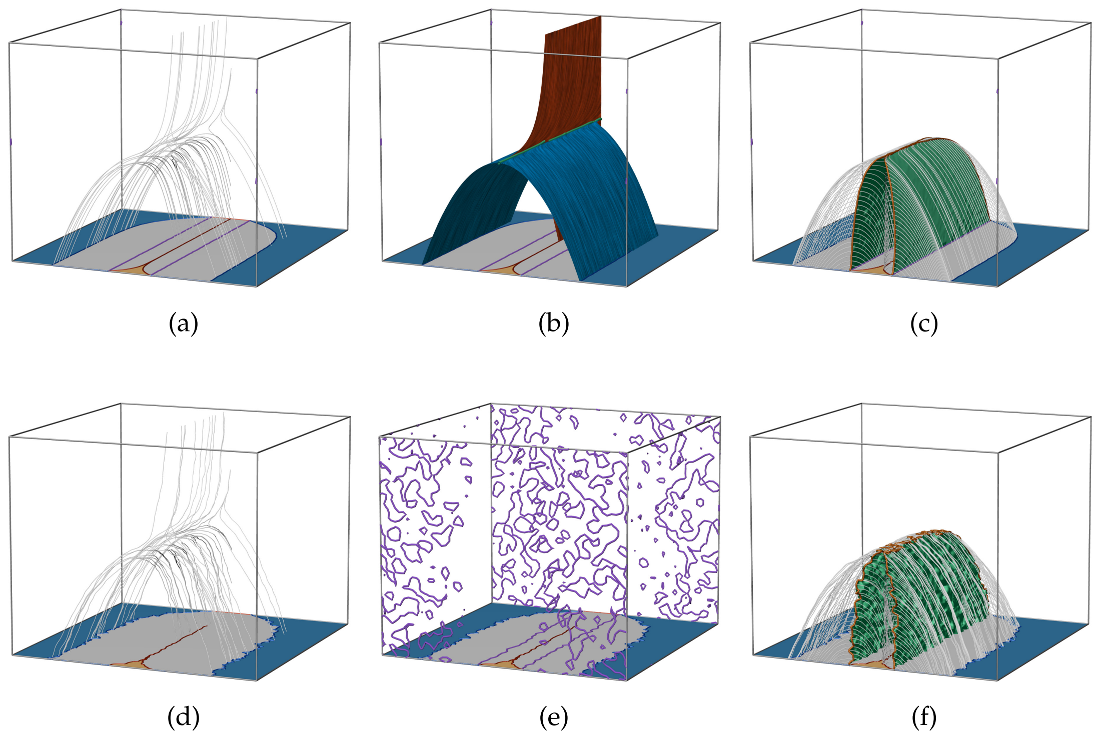


Figure 5.8 — Same as Figures 5.6 (a)–(c) but after applying 2% (upper row) and 30% of white noise (bottom row). The extracted bifurcation line is already disrupted at 2% (b), while our approach (c) still extracts the complete corridor. In case of 30% noise, the bifurcation line could not be extracted (e), which can indicate the structural stability of the bifurcation line. In this case, the isocline-based technique extracts only a spike, but still without disruptions, and consistent with streamline behavior (f).

sufficient to start field lines at the seeding curve and to detect if neighboring lines diverge too far from each other, i.e., if their distance exceeds δ. If this is the case, additional seeds are inserted between the respective field lines, these seeds are collected for efficient parallelization, and computed in a next pass on the GPU. Deletion of trajectories is employed if they become too close to each other as an optimization to improve computation time and to save memory with the resulting curves.

5.2.4 Isocline Surface Extraction and Trimming

The isocline surfaces are extracted using the Marching Cubes [Lorensen and Cline, 1987] algorithm on the subdivided grid. In the current implementation,

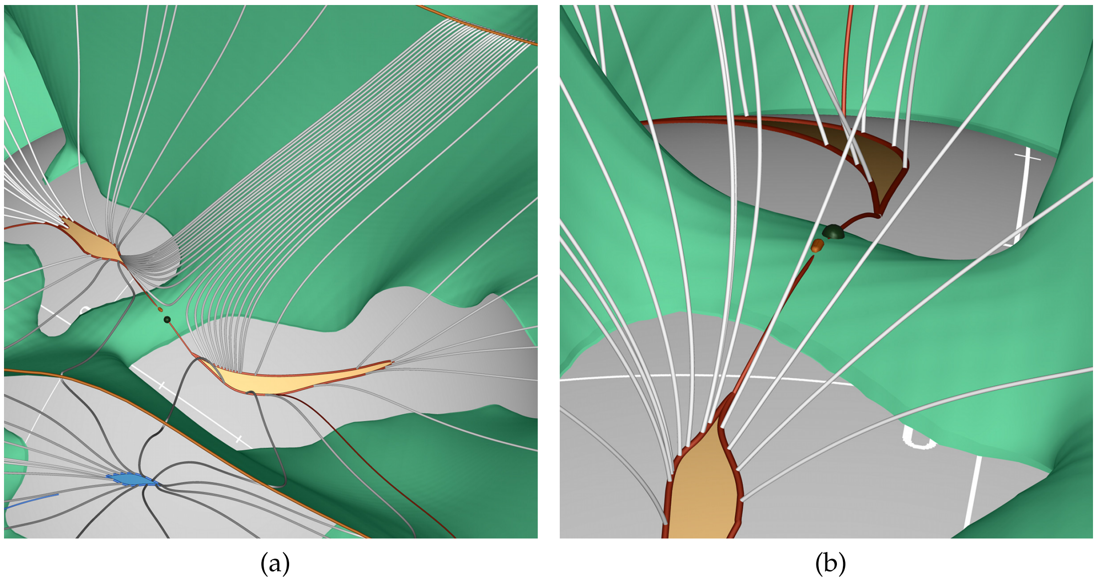


Figure 5.9 — MAS data at Carrington rotation 2131. (a) Two opposite boundary curve (orange) segments give rise to streamsurface parts (one illustrated by densely seeded streamlines) that partially converge to critical point (green point) along its 1D manifold (red curve). The difficulty in this case is that the resulting intersection curves with $\partial\Omega_M$ belong to two separate escape regions (yellow). (b) Same as (a) from another view. Since the critical point is located above $\partial\Omega_M$, the intersection curves are disrupted there. The 1D manifold connects the escape regions at their tips.

all cells that are initially occupied by the isosurface are subdivided to the same level. Because such a direct subdivision is computationally expensive, it is not employed too many subdivision levels. After the triangle mesh from these subdivided cells has been obtained, adaptive refinement of the triangle mesh is performed. Each triangle edge between two escape vertices is tested with recursive bisection for a rest streamline, and each triangle edge between two rest vertices is tested with recursive bisection for an escape streamline. If such a sample is found, the triangle is subdivided into four subtriangles and the process is iterated a predefined number of times. In a second pass, each triangle is tested for vertices intersected by opposite type of streamlines, i.e., it is tested if one gives rise to a rest streamline and the other to an escape streamline. If this is the case, the triangle is trimmed by repeating the recursive search and final cutting.

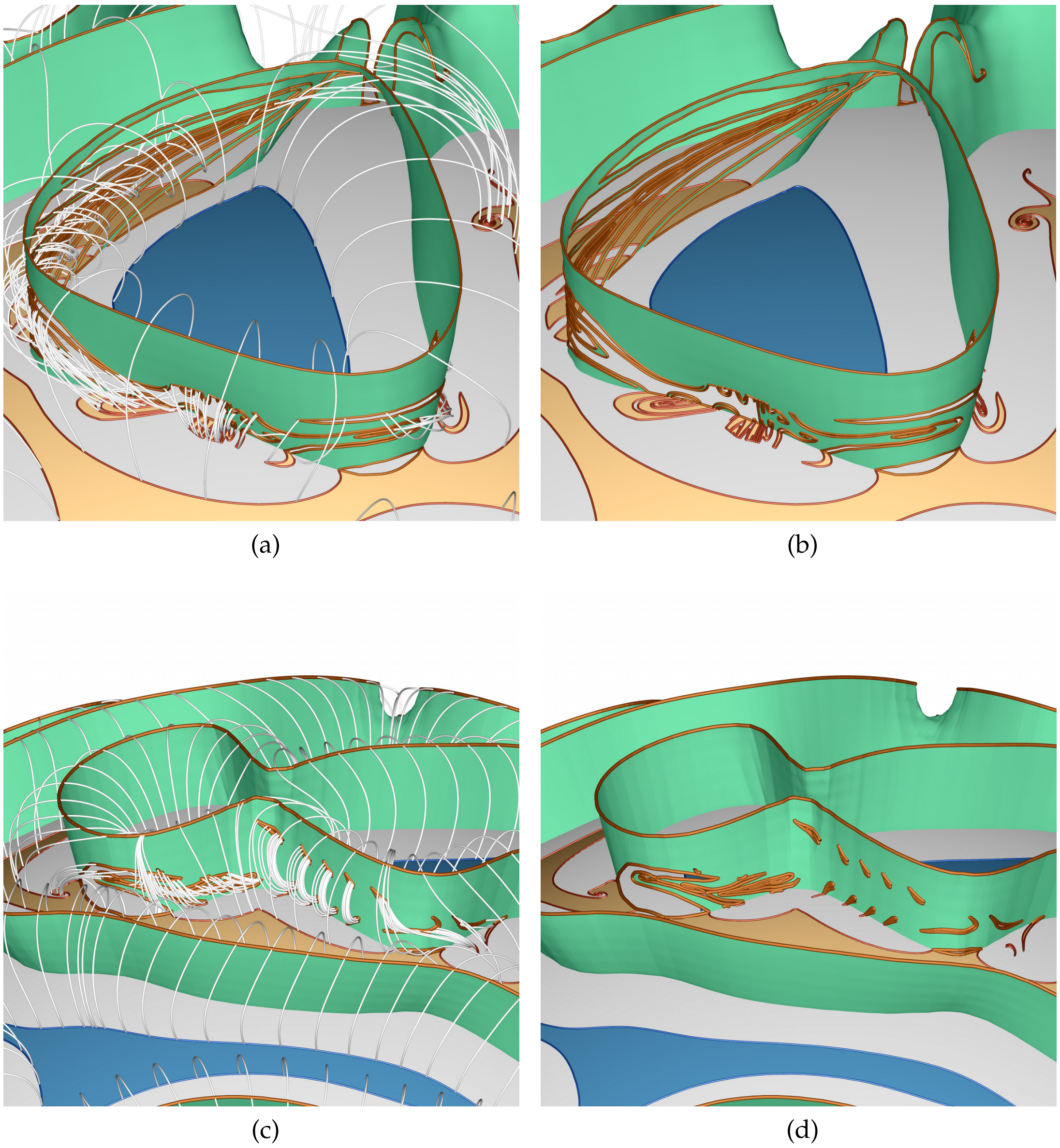


Figure 5.10 — Closeup of complex structures in the Magma Flow data that demand deep subdivision with the isocline-based algorithm for extraction (twice cell and six times triangle mesh subdivision). (a) The trimmed isocline surfaces present complex boundary structures (orange), which are caused by the large vortical flow crossing $\partial\Omega_M$, as one can see by the streamlines (white) sparsely seeded at the boundary curves for context. (b) Same as (a) but without streamlines to reveal the trimmed isocline surface, as well as the escape map. (c) The trimmed isocline surfaces present small boundary structures (orange), which are again caused by a vortical flow. (d) Same as (c) but without streamlines.

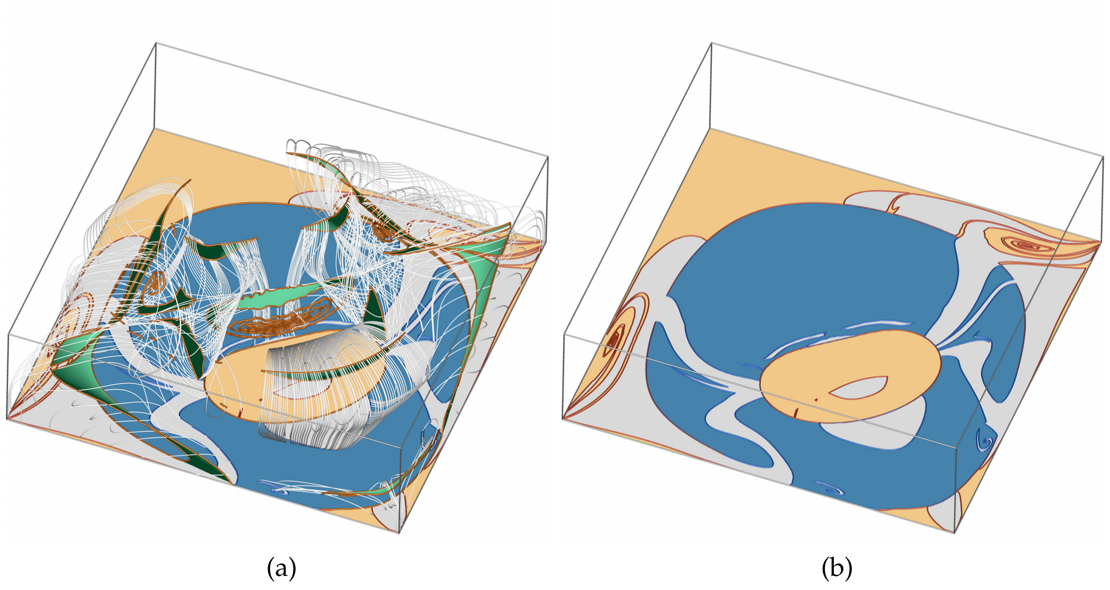


Figure 5.11 — Buoyant Flow CFD data. In this case $\partial\Omega_M$ is at the bottom face of Ω and the other five faces are $\partial\Omega_E$. (a) Trimmed isocline surface with seeding curves, some of the streamlines, and the resulting escape map. (b) Escape map visualizes which streamlines rise until $\partial\Omega_E$ and hence can transport heat from $\partial\Omega_M$ outside Ω.

5.2.5 Merging of Contours Using 1D Manifolds

As illustrated in Figures 5.9 (a) and (b), the streamsurface seeded at a seeding curve does not necessarily intersect $\partial\Omega_M$ along its full length, because there might be a saddle-type critical point leading to disruptions of the intersection curve between streamsurface and $\partial\Omega_M$. This is of no relevance to the visual result, as there will be an other streamsurface converging to the critical point from an other side and providing the missing part of the escape region boundary. These two boundary parts of the escape region converge at the spikes visible in Figure 5.9b, at the point where the 1D manifold of the critical point hits the escape region boundary. To obtain topologically connected boundary curves also in these cases, one detects if end points of escape region boundary parts are located closer than δ to an intersection between a 1D manifold and $\partial\Omega_M$. Hence, the 1D manifolds are used to decide which end points of escape region boundaries should be merged: they are only merged if the respective seeding curves are linked via the 1D manifolds to the same critical point. But, if the escape region boundaries are only needed for display, this step can be omitted, because the end points of the curves converge very close to each other.

5.3 Results

This technique is demonstrated on three examples with increasing computational complexity: (1) A buoyant flow based on a CFD simulation, (2) MHD-based coronal magnetic field data, (3) a flow simulation of the magma in the Earth's mantle. Table 5.1 provides some timings of the shown results. The experiments were performed on a system equipped with a GeForce GTX 560 Ti with memory capacity of 2 GiB.

Two levels of cell subdivision was used for the extraction of all isocline surfaces in our results. For the MHD data, mesh refinement was not employed, whereas, for the Magma Flow, six iterations of adaptive mesh refinement were further employed, and three iterations for the CFD data. Increasing these levels did not change the results visually (including the patterns visible on the isoclines, which reflect the trilinear interpolation of the data). For the trimming of the boundary triangles (Section 5.2.4), 50 bisection steps were employed. However, machine precision was typically achieved with less than 50 iterations. For critical point extraction, 20 subdivision levels were performed.

Table 5.1 — Performance measurements. Initial mesh and cell subdivision (init.), adaptive mesh refinement (adapt.), mesh boundary refinement (bound.), boundary advection (advec.), and total extraction time (total).

Data Source	**Time [s]**				
	init.	**adapt.**	**bound.**	**advec.**	**total**
Magma Flow	52.69	2930.55	5336.94	48.04	8371.46
Solar CR2131	515.84	0.52	193.05	410.80	1121.33
Solar CR2125	596.38	0.57	180.64	172.33	951.05
Solar CR2126	521.97	0.53	156.14	1264.22	1943.95
Solar CR2127	516.08	0.55	141.19	388.80	1047.73
Solar CR2128	585.21	0.61	179.77	410.81	1177.64
Buoyant Flow	30.67	309.01	965.78	10.75	1316.68
Analytic ($\lambda = 2$)	2.64	6.91	24.24	2.15	36.00
Analytic ($\lambda = 10$)	3.38	8.63	16.80	1.82	30.68

5.3.1 Buoyant Flow

The underlying dataset of the Buoyant Flow example is a CFD simulation of a closed container filled with air, which is heated at its bottom center and cooled at its top center, driving buoyant convection. The simulation grid consists of $61 \times 31 \times 61$ nodes arranged in a Cartesian grid, with extents $[0, 0.1] \times [0, 0.1] \times [0, 0.05]$. In accordance with traditional vector field topology, a single time step of the simulation is taken and is analyzed with the streamline-based escape map concept.

Figure 5.11 shows the result. Ω is set to $[0, 0.1] \times [0, 0.1] \times [0.01, 0.04]$, i.e., the map boundary is defined at $z = 0.01$ while the escape boundary reaches $z = 0.04$. Figure 5.11a shows the overall configuration of streamlines, isocline surfaces, and the resulting escape map. From the escape map in Figure 5.11b the following insights can be obtained. This analysis reveals a system of forward (blue) and reverse (yellow) escape regions. The center of the bottom wall, where the heated plate is located, is covered by both a reverse escape region and a forward escape region, while the forward escape region is larger. This shows that hot fluid leaves the bottom boundary in a rather large area while the cold flow that comes from the top reaches the boundary in a more focused region. This example demonstrates that escape maps could be useful for the analysis of flow separation and boundary-related flow phenomena in general. Assuming that the escape boundary would lead to, e.g., open flow, the escape map could provide information where hot fluid is leaving the bottom of the chamber and hence leads to energy loss. Using a streamline probe or a set of streamlines, it would be very difficult to locate the regions at $z = 0.01$ that exhibit different streamline behavior.

5.3.2 Solar MHD

Solar magnetic field data is used from the *Predictive Science Modeling Support for Helioseismic and Magnetic Imager Solar Dynamics Observatory* site [pre] in this example. The MAS data are available with a temporal granularity of one Carrington rotation (CR), which is the period for a complete rotation of the Sun, which is about 27 days, starting from November 9, 1853. The data are given on a spherical staggered grid with resolution $181 \times 100 \times 150$ that exhibits increasing cell size in radial direction with increasing distance from the photosphere. In this implementation, the staggered grid was interpolated to a structured node-based spherical grid of the same resolution. With this conversion the implementation of special streamline integration code that operates on staggered grids is avoided, and the visualization of the field using standard techniques is also eased. The lower radial boundary of the MAS data resides slightly below the photosphere while the upper boundary is located at about $30 \cdot R_{\odot}$ (solar radii) and so does the grid of the converted data. Throughout this implementation, trilinear interpolation is used consistently, both for the magnetic field (e.g., for streamline integration) and its radial component (e.g., for isocline surface extraction).

The original idea of escape maps comes from the determination of coronal

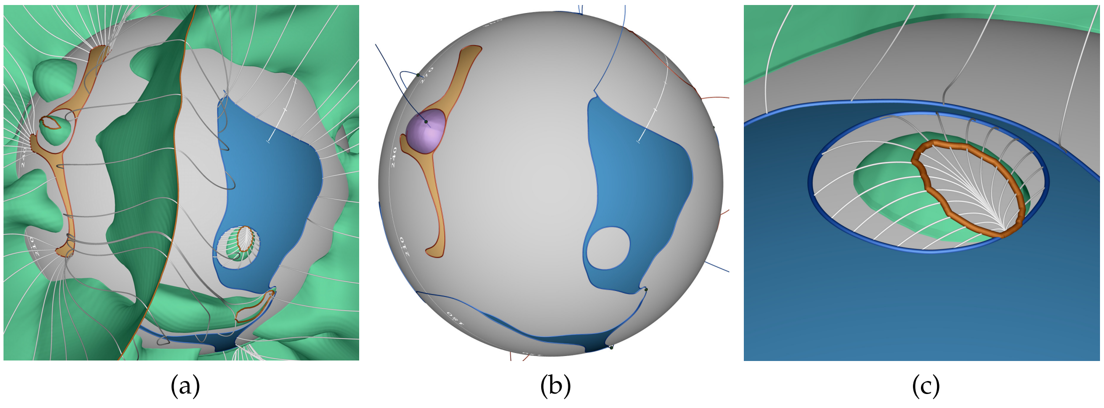
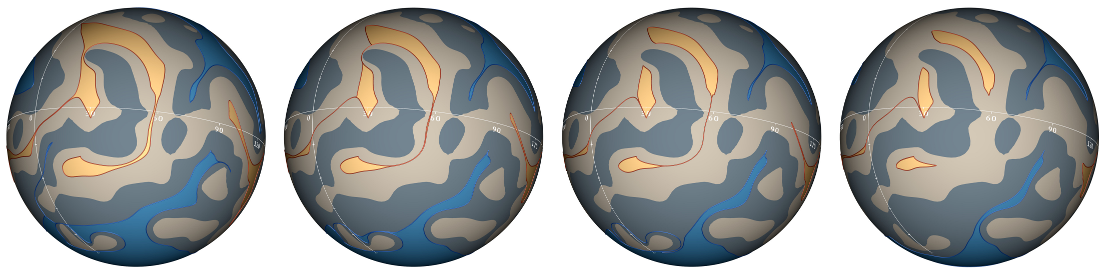


Figure 5.12 — MAS data at Carrington rotation 2131. (a) Trimmed isocline surface with seeding boundary curves (orange), some of the resulting field lines (white), and the resulting coronal hole boundaries (red, blue). (b) Only a single boundary curve segment is contributed by the topology-based approach (2D manifold in transparent pink together with 1D manifolds). Note that the 2D manifold does neither reach $\partial\Omega_M$ everywhere, nor does it reach $\partial\Omega_E$ (Figure 5.4c), instead it contributes a geometrically open segment. (c) Closeup of 'hole' in blue region. This represents the rather rare case of Figure 5.4d (ii) covered by Step 4. in Section 5.1.2. The 'purple' segment (boundary switch curve) is the part of the orange boundary curve at $\partial\Omega_M$.

holes (Section 4.3.7). In this case, the map boundary $\partial\Omega_M$ is identified with the photosphere, a sphere of radius $R_\odot$, which is defined as the visible surface of the Sun. The escape boundary $\partial\Omega_E$ is set to a sphere of radius $R = 2.5R_\odot$. Then, escape regions mark the coronal holes.

In Figure 5.1 the MAS data from Carrington rotation 2128 is visualized. Figure 5.1a shows the traditional approach, i.e., by view-dependent direct sampling. From this picture, it is not possible to judge which of the regions (i), (ii), (iii), and (iv) are connected. As shown in Figure 5.5a, the topology-based extraction step provides a single curve on this side of the Sun for this dataset, while Figure 5.1b shows the trimmed isocline surfaces with some of the field lines seeded at their boundary curves, which generate the major part of the boundary curves of coronal holes, resulting in Figure 5.1c. Finally, Figure 5.5b shows that (i), (ii), (iii), and (iv) are located within the same region of 'north' polarity but represent three distinguished coronal holes. Note that the boundaries of unipolar regions are identical to the $\partial\Omega_M$-boundary of the isocline surfaces and identical to boundary switch curves. Due to this fact, coronal hole boundaries cannot pass boundaries of the isocline surfaces or boundary switch curves.



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

Figure 5.13 — MAS data at Carrington rotation 2127. Coronal hole boundaries and polarity of the photosphere (blueish 'north', yellowish 'south') for escape radii $R_e = 1.2R_\odot$ (a), $R_e = 1.35R_\odot$ (b), $R_e = 1.5R_\odot$ (c), and $R_e = 2.5R_\odot$ (d). More disconnected coronal holes per unipolar region are obtained with large R_e and that also the genus of the coronal holes can change (the closed corridor at the bottom disappears between (c) and (d)).

The regular artifacts visible on the boundary curves of trimmed isocline surfaces (e.g., Figure 5.12c) and on the 2D manifolds (e.g., Figure 5.5a) are due to the trilinear interpolation scheme of the original data. The same holds for the visualization (e.g., Figure 5.5b) which is achieved by view-dependent sampling using trilinear interpolation. These interpolation effects have no impact on the technique because the same interpolation is used in all steps, i.e., from the visualization point of view, the trilinearly interpolated field represents the true field. This consistency is also illustrated by the fact that although the seeding curve in Figure 5.12c is not smooth, the resulting intersection curve (blue) exhibits a smooth shape.

Figure 5.12 shows the results for the MAS data from Carrington rotation 2131. Figure 5.12a presents a visualization similar to Figure 5.1b. There are two 'holes' in the escape regions (coronal holes), one to the left of the central isocline surface and one to the right. However, these two similar 'holes' are extracted differently. The left one is obtained by intersecting a 2D manifold of a saddle point, as shown in Figure 5.12b. This is again the only 2D manifold that contributes a boundary curve on this side of the Sun. In contrast, the right 'hole' is generated by the isocline surface-based approach, as shown in Figure 5.12c.

Figures 5.9 (a) and (b) show Carrington rotation 2131. Figure 5.9a illustrates two issues: the divergence of a streamsurface because it hits a critical point along its 1D manifold, and the related fact that therefore a single segment on the isocline boundary seeding curve can give rise to streamsurface parts that are mapped to two distinct escape regions.

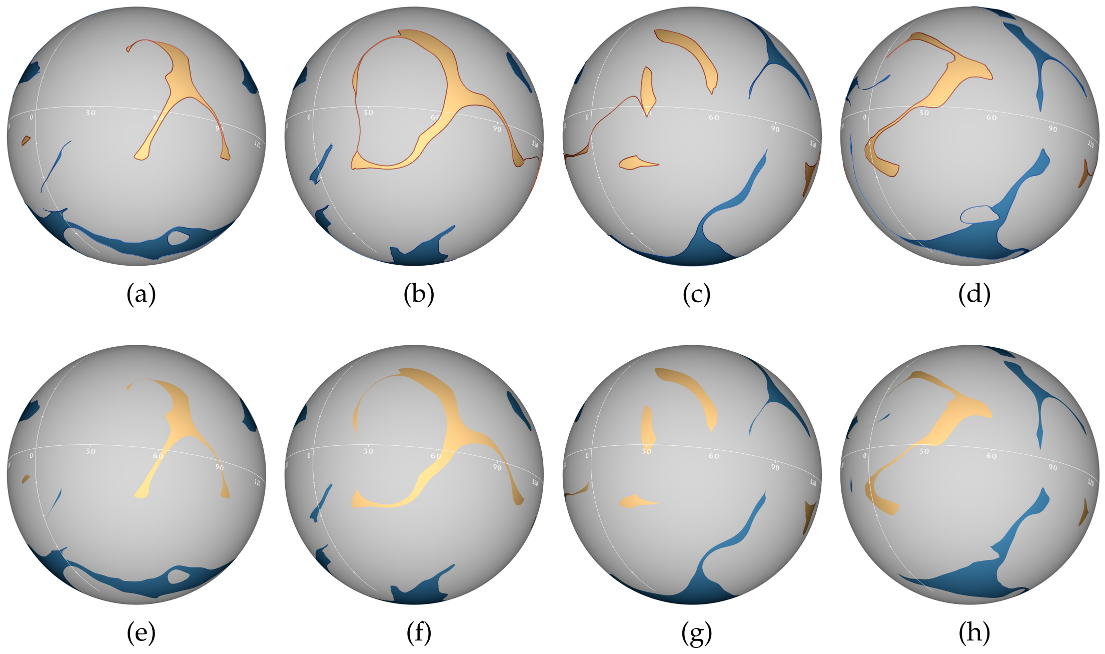


Figure 5.14 — MAS data for Carrington rotations 2125 (a), 2126 (b), 2127 (c), and 2128 (d). Due to the clear representation of the corridors by the isocline-based technique, one can easily observe topological changes of the coronal holes. See Figure 5.13 for the polarity of the photosphere in case of (c). (e)–(h): Same as (a)–(d) but with traditional direct sampling approach. It is apparent that, again, this approach is not able to resolve the corridors.

Figure 5.14 provides a clear picture on the morphological and topological changes of the coronal holes over four Carrington rotations, in particular regarding the appearance and disappearance of corridors. Figures 5.14 (e)–(h) provide a comparison with the traditional visualization technique, which fails at resolving the thin corridors.

Morphological studies of coronal hole boundaries as observed in the soft X-ray regime by the Yohkoh telescope imply that, in some unipolar regions, coronal holes may consist of several, apparently disconnected, components (see, e.g., Kahler and Hudson [2002]). Antiochos' uniqueness conjecture of coronal holes, however, states that "coronal holes are unique in that every unipolar region on the photosphere can contain at most one coronal hole" [Antiochos et al., 2007; Titov et al., 2011]. The apparent disconnected coronal holes were either connected by a thin, but finite, corridor, or they are linked by "singular lines of zero width".

In the experiments of this work, cases where the isocline-based technique

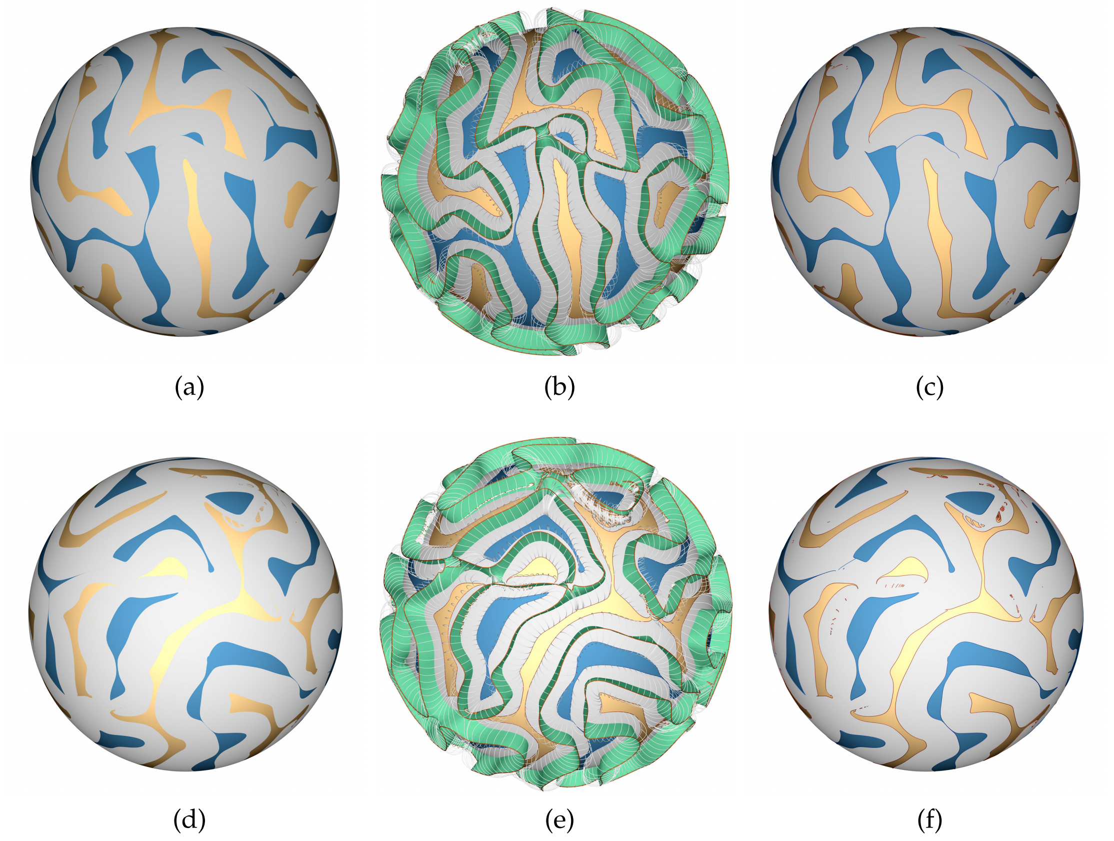


Figure 5.15 — Escape maps of Magma Flow dataset from two different viewpoints (top and bottom rows). $\partial\Omega_M$ and $\partial\Omega_E$ are positioned at radii $R_M = 0.8R_\oplus$ and $R_E = 0.9R_\oplus$, respectively. The map indicates regions at $R_M = 0.8R_\oplus$ from where the flow reaches $R_E = 0.9R_\oplus$ in forward (blue) and backward (yellow) direction. Left ((a) and (d)) the visualization with the view-dependent direct sampling approach is shown. This approach suffers from sampling issues and cannot capture the full topological structure of the escape regions. Note that it is not possible to judge the connections between the apparent escape regions. The center ((b) and (e)) shows the isocline-based approach, see Figure 5.10 for closeups. Right ((c) and (f)) shows the resulting escape map, which according to this approach provides the full topological structure.

extracted more than one disconnected coronal hole in a unipolar region were identified, one such case is shown in Figure 5.5b. The influence of the choice of the escape radius R_e ($\partial\Omega_E$) was examined. Figure 5.13 shows a respective result for Carrington rotation 2127. One can observe changes of the coronal holes with respect to shape and topology. For example, the coronal hole located at the image center at $R_e = 1.2R_\odot$ thins down as R_e increases and represents a corridor there at $R_e = 1.5R_\odot$, before it vanishes for $R_e = 2.5R_\odot$. Using $R_e > 2.5R_\odot$ did not result in further substantial changes of the coronal holes. This contradicts Antiochos' conjecture. However, detailed investigations have to be subject of future research in the field of astrophysics and scientific visualization.

5.3.3 Magma Flow

The Magma Flow dataset was obtained using [cit], which is a finite element code designed to solve compressible thermochemical convection problems relevant to Earth's mantle. In contrast to the MHD dataset, the magma flow presents more complex structures like large vortical flows convecting magma between deep and surface-proximate Earth mantle regions. The analysis of the dynamics of this flow and a mapping of mass transport between the different levels of depth inside the Earth can support the understanding of volcanic processes and the movement of plate tectonics, which can also facilitate studies about prediction of volcanic eruption or earthquakes. Here, a global simulation domain is used given on a spherical grid of resolution $315 \times 157 \times 24$. Aiming to capture special cases of high complexities for this algorithm, and to avoid the high computation time of extracting subdivided isocline surfaces for such a large and complex dataset, the map and escape boundaries are setup as two spheres of radii $R_M = 0.8R_\oplus$ (earth radii) and $R_E = 0.9R_\oplus$. In this configuration, some cases of vortices intersecting $\partial\Omega_M$ could be captured, for example, the cases shown in Figures 5.10 and 5.15. To this dataset, cell subdivision and adaptive mesh subdivision with depths two and six, respectively, have been applied. For integration, a maximum of three thousand steps of size 0.0005 were performed. Note that this dataset has higher resolution than the Buoyant Flow dataset and it also needs even deeper mesh subdivision. Both yields in a computation time of about two and a half hours for the extraction of its trimmed isocline surfaces.

Chapter 6

Image-Based Streamsurfaces

Streamlines are integral curves (see Section 2.3) everywhere tangent to a steady vector field (or an isolated time step of time-dependent fields), and are defined by seed points, i.e., exactly one streamline passes through a seed point. A streamsurface is an extended concept and consists of an infinite set of streamlines that pass through a given seed curve (see Section 2.3.1). Beyond showing direction, streamsurfaces provide insights on flow behavior by providing structures that reveal properties of, e.g., swirling behavior or flow separation. In traditional vector field topology, streamsurfaces are an essential tool for computing two-dimensional separatrices that arise from saddle-type critical points or saddle-type periodic orbits. Streamsurfaces are also used to compute separatrices from further constructs such as boundary switch curves (Section 2.5.4) and the bifurcation lines presented in Chapter 3. Moreover, streamsurfaces can be used as building blocks for other methods [Garth et al., 2004; Theisel et al., 2003].

Streamline computation is typically performed by explicit integration of a vector field, which represents a straightforward task. Streamsurface computation, on the other hand, may lead to expensive computational efforts and possibly considerable memory usage. Most of the previous techniques for computing streamsurfaces focus on finding a triangle mesh approximation to the surface [Hultquist, 1992; van Wijk, 1993; Garth et al., 2008; Krishnan et al., 2009; Stöter et al., 2012; Schulze et al., 2012]. Such a triangle mesh at hand can feature some advantages, like presenting fast rendering after extraction and supporting subsequent processing. The extraction of high quality meshes is, however, still challenging and frequently expensive. Moreover, extracting and storing such surfaces may be costly and unnecessary application-dependent, and searching for specific streamsurfaces (e.g., by interactively defining seed curves) can turn out to be a tedious task. Therefore, for many situations it is desirable to apply a fast method to compute streamsurfaces for direct visualization.

Aiming to avoid the triangulation cost, Schafhitzel et al. [2007] presented a point-based approach, which is the most closely related to the technique presented in this chapter. They sample a dense set of points along the streamsurfaces and apply splatting to avoid noticeable gaps on the visualization. Although, they achieved interactive frame rates in some cases, their approach demands user-assisted parametrization to prevent noticeable gaps. Moreover, they pre-compute all particles for further rendering, and, therefore, may demand high memory availability, for example, in complex flow configurations exhibiting high divergence.

This chapter presents an image-based technique that renders streamsurfaces without performing triangulation as well. This approach renders dense sets of streamlines directly during explicit integration of the streamsurfaces. Its data structure keeps in memory only data referring to the front of the last two time steps of integration. Different to the point-based approach presented by Schafhitzel et al. [2007], this technique does not require user parametrization to prevent noticeable gaps, allowing for uninterrupted exploration. The GPU-based implementation shows that this method attains interactive frame rates for reasonable screen resolutions and reasonable flow complexity. Moreover, LIC [Cabral and Leedom, 1993] is incorporated to the implementation, which demonstrates the technique's suitability with existing texture-based flow visualization methods on surfaces.

6.1 Physical-Space Streamsurfaces

The technique presented in this chapter can be classified as an explicit approach (see Section 2.3.1) and is based on Hultquist's approach [Hultquist, 1992]. As such, it can be parametrized in p-space as follows. Given a steady vector field $\mathbf{u}(\mathbf{x})$, this chapter uses the parametric representation of a streamsurface as the function of position $\mathbf{x}(\rho, \tau)$ of particles ρ along the respective streamlines over time τ. Thus, streamsurfaces are computed by solving the ODE:

$$\frac{\partial(\mathbf{x}(\rho, \tau))}{\partial \tau} = \mathbf{u}(\mathbf{x}(\rho, \tau)). \tag{6.1}$$

In a continuous representation, $\rho \in [\rho_0, \rho_1]$ and $\tau \in [\tau_0, \tau_N]$. The seed curve is hence represented by $\mathbf{x}(\rho, \tau_0)$ and streamlines by $\mathbf{x}(\rho_i, \tau)$, where $0 \leq i \leq 1$.

Furthermore, a front, which according to this representation is a timeline, is hence $\mathbf{x}(\rho, \tau_c)$ for any constant time τ_c.

The algorithm for explicit computation of streamsurfaces, as presented in this chapter, starts by placing particles along the seed curve $\mathbf{x}(\rho, \tau_0)$, which represents the front at time $\tau_c = \tau_0$. This sampling is performed under user constraints, where the distance between consecutive sample particles in physical space must not be larger than a user-defined δ_{max} and should not be smaller than a user-defined δ_{min}. Here, in order to simplify the algorithm for resampling fronts, δ_{max} is kept as a required constraint and δ_{min} as just "desirable". All sample particles are then integrated by a user-defined time step ω, which gives a new front at $\mathbf{x}(\rho, \tau_0 + \omega)$. Notice that, in flow visualization, the value of ω is typically chosen a fraction of the cell size at that position of the vector field's grid, but in this formulation a constant time step is here assumed to simplify the notation. The algorithm then resamples the particles at $\mathbf{x}(\rho, \tau_0 + \omega)$ according to δ_{min} and δ_{max}, i.e., it removes or inserts particles accordingly. It performs further iteratively as long as $\tau_0 + s \cdot \omega \leq \tau_N$, where s is the number of time steps already employed. Finally, if $\tau_0 + (s + 1) \cdot \omega \geq \tau_N$, then the last time step needs to be employed as $\omega_{last} = \tau_N - (\tau_0 + s \cdot \omega)$.

6.2 Image-Based Streamsurfaces

This chapter presents an approach, whose basic idea is to avoid the triangulation of the streamsurface. Hence, the streamsurfaces are rendered with just lines or points. To do so, the approach described in Section 6.1 needs to be adapted in an image-based sense, involving image-space parameters that are analogous to the physical-space δ_{min}, δ_{max}, and ω. In fact, at least the physical space parameters δ_{max} and ω are kept for global consistency, i.e., the image-space parameters are employed only if they do not violate these physical space constraints, because otherwise they could affect the quality of the resulting surface. For example, this could happen if long p-space step sizes are used or substantial errors are introduced by resampling the fronts based on existing samples that are at a "long" distance from each other.

6.2.1 Image-Based Front Resampling

Resampling of the streamsurface fronts, as aforementioned, requires the two p-space parameters δ_{min} and δ_{max}. The value of δ_{max} is typically chosen such that it limits the error introduced by the insertion of particle samples during front resampling. As this error potentially grows during line integration, its estimation is usually unpractical. Therefore, a relatively small value should be chosen. On the other hand, excessive sampling, caused by flow convergence, may vainly affect computational performance. Thus, δ_{min} is selected sufficiently large, causing sufficient particle removal.

The image-based resampling of the streamsurface fronts consists of finding a set of particle samples along each front, such that the distance between them is maintained according to the image-space parameters δ'_{min} and δ'_{max}. These parameters represent distances in image space, i.e., relative to the Picture Element (pixel) size, which is assumed throughout this chapter to be equal to one. Assume that $\mathbf{x}'(\rho_i, \tau_t)$ is the position of $\mathbf{x}(\rho_i, \tau_t)$ in image space. The physical-space parameter δ_{max} is also used in this image-space approach to control error growth. Thus, N evenly-spaced particle samples are inserted to the front $\mathbf{x}(\rho, \tau_t)$ between the consecutive samples $\mathbf{x}(\rho_i, \tau_t)$ and $\mathbf{x}(\rho_j, \tau_t)$, where

$$N = \left\lfloor \max\left(\frac{\Delta(\rho_i, \rho_j, \tau_t)}{\delta_{\text{max}}}, \frac{\Delta'(\rho_i, \rho_j, \tau_t)}{\delta'_{\text{max}}} \right) \right\rfloor, \tag{6.2}$$

and $\Delta(\rho_i, \rho_j, \tau_t)$ is the distance between $\mathbf{x}(\rho_i, \tau_t)$ and $\mathbf{x}(\rho_j, \tau_t)$ in physical space, and $\Delta'(\rho_i, \rho_j, \tau_t)$ is their distance in image space. Particle $\mathbf{x}(\rho_i, \tau_t)$ is removed if its image-space distance to its previous $\mathbf{x}(\rho_h, \tau_t)$ and next neighbor $\mathbf{x}(\rho_j, \tau_t)$ is smaller than δ'_{min}, if and only if this removal does not conflict with Equation 6.2.

6.2.2 Image-Based Line Integration

The time step ω for line integration is typically selected such that it is small enough with respect to integration error. The precision of higher-order integration schemes, like the Runge-Kutta Fourth-Order Method (RK4) or Dormand-Prince Fifth-Order Method (DOPRI5), is often satisfactory when the time step corresponds to about one fifth of a cell size or smaller.

In an optimal situation, image-based line integration would be parametrized such that an image-based step size ω' limits the physical-space time step ω, i.e., an integration step is employed with ω, unless ω' would require a smaller step. However, mapping ω' to an optimal step size in physical space is not straightforward and in practice ω' can only be determined after the step was employed. Thus, the most straightforward approach would be to employ the step in physical space, as usual, with ω. Afterwards, one could check the step length in image space and, if necessary, interpolate a position along the step employed, accordingly. This approach, however, would only be suitable for explicit Euler integration, and could not be applied for higher-order schemes like RK4 or DOPRI5. For such cases, one could employ a bisection method on

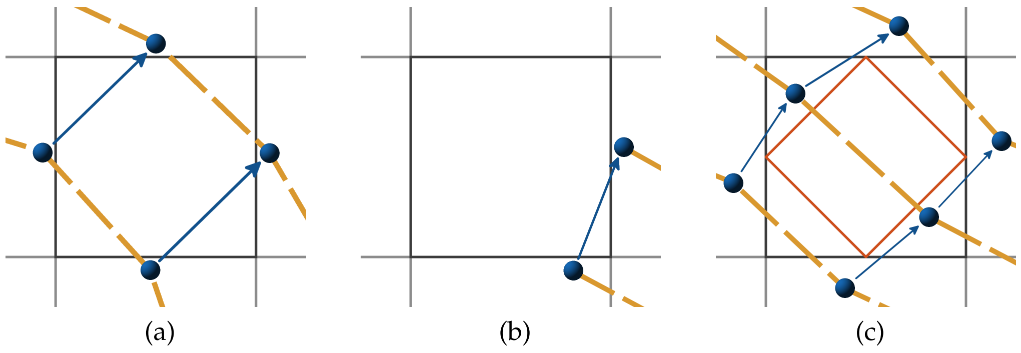


Figure 6.1 — Illustrations of image fragments (black square). In image space, the particles (blue spheres) are sampled along fronts (yellow dashed curves) and propagated along the flow field with an integration method (blue arrows). (a) Choosing parameters $\delta'_{\max}$ and ω' greater than $\frac{1}{\sqrt{2}}$, even if smaller than the fragment's size, makes the approach for rendering particles susceptible to visible gaps. (b) Border cases, which happen when the fragment is at a border between surface and non-surface regions on the image. These cases can be neglected in this approach without impact on the quality of the rendered surfaces. (c) The diamond-exit rule, which demands that a parametrization of $\delta'_{\max}$ (or ω') also equals to $\frac{1}{\sqrt{2}}$ if rendering lines.

the step size in p-space to find a satisfactory value for ω that satisfies ω', too. Even with the cache coherence of currently available graphics hardware, this approach would, however, strongly impact on the technique's performance by demanding many reads from the vector field due to the iterative character of the bisection method. Moreover, it could also in worst cases not be able to find a satisfactory step size at reasonable iteration count, i.e., a step size in physical space that leads to a step that is smaller than ω' in image space. Therefore, a conservative approach that maps the length of ω' from image space directly to physical space is adopted, as if it was the length of a vector parallel to the image plane, located at the position (depth) of the integration. The actual step length employed is hence chosen as the smallest among the mapped length and ω. Thus, this approach limits the step size in image space and in physical space to a maximum of ω' and ω, respectively.

6.2.3 Method

With the image-based front resampling and image-based line integration at hand (Sections 6.2.1 and 6.2.2), three different approaches for image-based streamsurfaces are here analyzed. They basically differ from each other by the rendering method: *particle-based*, *streamline-based*, and *front-based* rendering. The techniques described in the following sections assume the Open Graphics Library (OpenGL) specifications for rasterization.

Particle-Based Rendering

This approach represents the most brute force among the ones presented in this chapter. As such, it is assumed that the surface rendering will be performed in a point-based fashion and for visual quality points with size one are assumed, i.e., no splatting is applied and one particle sample will render at most one pixel on the screen. The aforementioned image-based methods for particle sampling along fronts (Section 6.2.1), and line integration (Section 6.2.2), must be combined with each other in such a manner that every image fragment in the output that is passed by the streamsurface during integration must retain at least one particle sample for rendering. Hence, a robust parametrization (δ'_{min}, δ'_{max}, and ω') must be defined to fulfill this requirement.

According to the graphics API, due to a given input particle $\mathbf{x}(\rho_i, \tau_t)$, if $\mathbf{x}'(\rho_i, \tau_t)$ is inside the viewport, then the fragment at position

$$(x, y) = \left(\left\lfloor \mathbf{x}'_x(\rho_i, \tau_t) \right\rfloor + 0.5, \left\lfloor \mathbf{x}'_y(\rho_i, \tau_t) \right\rfloor + 0.5 \right) \tag{6.3}$$

must be rasterized. Therefore, a fragment consists in image space of a square with side length equal to one, whose area includes the left and top edges and excludes the bottom and right edges. If a particle's position in image space is inside this area, then this fragment is selected for shading. Hence, to define the values of δ'_{max} and ω', border cases are neglected, which consist of fragments at borders between surfaces and non-surface regions in the image, due to the current integration step. Figure 6.1b illustrates such a border case. In this example, the particle (blue spheres) at open end points of the fronts (yellow dashed lines) employs one integration step (blue arrow) that crosses the fragment area without providing a sample inside it. Defining a parametrization that handles these cases demands very small values, and incorporating methods to detect these cases would represent an unnecessary effort, because the border cases can be neglected without impacting the quality of the rendered surface. For all other cases, and to avoid perceivable gaps in the rendered image, the side length of the inscribed square of the incircle of one fragment is selected, i.e., $\delta'_{\text{max}} = \omega' = \frac{1}{\sqrt{2}}$. Any greater value for any of the two parameters would make the approach susceptible to perceivable gaps on the resulting surfaces as illustrated in Figure 6.1a, where greater values for the parameters were chosen and the fragment at the center would be missed by the approach.

Streamline-Based Rendering

Aiming a method that is more efficient, one may choose to render lines instead of points. By rendering streamlines the integration constraints can be relaxed, i.e., employing line integration with just the physical space parameter ω and applying the image-based method only for the front resampling. Now it has to be ensured that whenever two consecutive streamlines are rendered, there are no gaps between them.

The rasterization of lines adopts the *diamond-exit* rule, which consists of a kind of Bresenham's algorithm. According to the diamond-exit rule, if the line intersects a diamond-shaped area (red square in Figure 6.1c) defined by $D = \{(x, y) \mid |x - cx| + |y - cy| < 0.5\}$, where (cx, cy) is the center of the fragment, then the fragment is selected for shading, except for line end points residing inside this area. Thus, to avoid gaps, $\delta'_{\max}$ is set to the length of one side of this diamond shaped-area, which is $\delta'_{\max} = \frac{1}{\sqrt{2}}$, as well.

Front-Based Rendering

Another possible approach for a line-based rendering is to render streamsurface fronts instead of streamlines. In this case, image-based line integration is employed together with physical-space front resampling. By adopting an analogous logic as for rendering streamlines (e.g., switching fronts by streamlines in Figure 6.1c), $\omega' = \frac{1}{\sqrt{2}}$ is used.

6.2.4 Discussion

Typically, the parameters $\delta'_{\max}$ and ω' restrain the physical space parameters $\delta_{\max}$ and ω. Therefore, the particle-based rendering approach is potentially very accurate. It performs very fine sampling of the surface and its accuracy is, in fact, limited by the output image's resolution. Its computation is, however, more expensive than the other two rendering approaches. It usually performs as many integration steps as the front-based rendering approach and consequently as many front resamples. The front-based rendering allows for front resampling after more than one integration step without producing visual gaps. This is, however, not recommended, because in the experiments it was observed that it can affect surface quality. Moreover, the task of resampling fronts with the particle-based approach is more expensive than with the front-based, because the particle-based approach tends to perform over more elements along the front at each resampling. The data structure for both approaches demands to maintain the last front during integration and drawing, which is performed simultaneously in this technique. The streamline-based rendering approach,

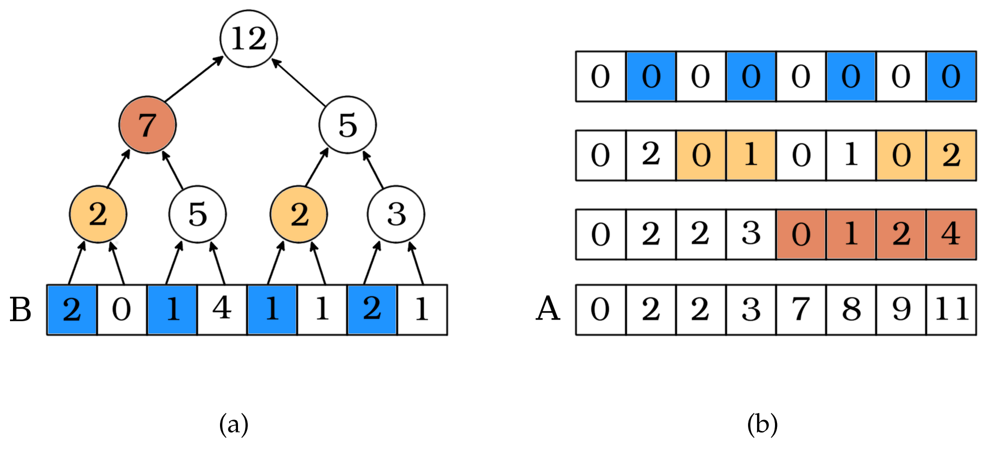


Figure 6.2 — Illustration of the fast resampling algorithm (Sec. 6.3). The array B is positioned at the leaves of the tree (a). Each node of this tree is then assigned the sum of its two child nodes. (b) The values of A during the iterations of our algorithm (top to bottom). This algorithm can run in parallel by assigning a separate thread for each one of the elements of A, with a synchronization stop between iterations.

on the other hand, usually employs less steps than the other two approaches and, although each front resampling is performed over as many elements as for the particle-based rendering, the streamline-based rendering is the approach that usually computes faster. Its data structure keeps at least the last two fronts during integration and drawing, which is not critical, because this technique has low overall memory usage. And while performing front resampling, one must consider the constraints over the values from the previous front, as well. The value of δ'_{min} should be selected to balance memory capacity and computation time. In the experiments, it was satisfactorily set to $\delta'_{\text{min}} = \frac{2}{3} \times \delta'_{\text{max}}$.

6.3 Fast Front Resampling

The front resampling is a bottleneck of the streamline-based algorithm. Typically the fronts are resampled once for each integration step, except for the front-based rendering that can perform front resample after several integration steps. Here, the data structure is represented as arrays of particles to achieve faster primitive rendering. However, inserting and removing particles from arrays requires resizing the arrays many times in this technique. A straightforward implementation of the front resampling would basically consist of iterating over each element of the array, copying the particle data to a second array while inserting new ones when required. After each iteration, the arrays would be switched for rendering. Notice that this algorithm has complexity of the order $O(n)$. The technique presented by Schafhitzel et al. [2007] approaches this particle connectivity with linked lists on the GPU, which eases the tasks of inserting and deleting particles. However, linked lists are not compatible with typical rendering primitives. Hence, one must convert this data structure to array representation for rendering, which impacts on the performance of the technique.

This chapter proposes an implementation that, by adopting a divide and conquer strategy for multiprocessors (e.g., in the experiments a GPU implementation was used), exhibits complexity $O(\log(n))$. This algorithm splits the resampling into two steps. First, the output array indices are mapped for storing the new and remaining particles from a given input front sampled in the array $P_{in} = \{p_0, p_1, \ldots, p_{N-1}\}$ of size N, where p_i represents the particle samples. The data structure for this mapping is hence stored in two further arrays $A = \{a_0, a_2, \ldots, a_{N-1}\}$ and $B = \{b_0, b_1, \ldots, b_{N-1}\}$ with same length. Then, P_{in} is processed to store in A the indices j at which particles descendant from P_{in} will be copied to at the output array P_{out}. In B, the number of particles descendant from each element of P_{in} is stored. For example, if one needs to insert m particles between p_i and p_{i+1} (m includes p_i), then $b_i = m$ and a_i stores the index j at which the first from these particles will be written to. Hence, if $m = 0$ then the particle sample p_i is removed. Afterwards, once P_{in}, A, and B are properly set, one can build the appropriate output array P_{out} in parallel in almost constant time, depending on the number of available processors, by assigning one thread for each element of P_{in}.

Finding the values of A and B starts by computing B, which is straightforward and can be achieved in parallel. However, because front resampling is here performed in parallel, it is not trivial to decide which elements from P_{in} are optimal candidates for removal (e.g., in a sequence of particles that are all very close to each other in physical and/or image space). Moreover, removing all particles with closer distances than the minimum constraints ($\delta_{\min}$ and/or $\delta'_{\max}$) only with a local test can lead to large distances between consecutive particles, due to excessive removals, which may conflict with the maximum constraints ($\delta_{\max}$ and/or $\delta'_{\max}$). A simple solution is, hence, to remove a given particle p_i only if it is at an odd location on the array, i.e., if i is odd.

For computing the array A, B is taken as input. Then, a binary tree T is used, where each node of T stores the sum of its two child nodes. The tree is built by assigning the values at the leaves with the content of B. Figure 6.2 illustrates one example for this algorithm where (a) illustrates T and (b) shows the values

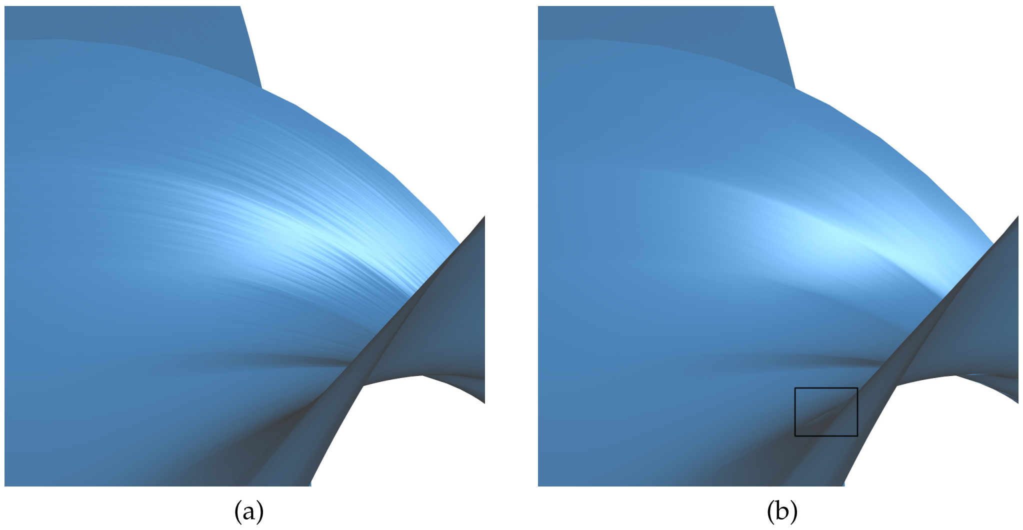


Figure 6.3 — Zoomed view showing part of the streamsurface from Figure 2.2b, rendered without LIC. Results of our approach without (a) and with (b) tangent vector advection. Note that adopting the tangent vector advection makes the shading insusceptible to front deformation and thus produces better shading. However, this approach does not offer full control over the tangents computed, therefore some small artifacts may become visible, for example, light blue artifacts near torsion regions (box) (b).

of A after each iteration from top to bottom. A is hence initialized with zeros. The algorithm starts at the leaves and iterates upwards on T. At each iteration, for each node f of T at the ascendant level with left child node l and right child node r, the elements of A that reference children of l retain their values, while the elements of A that reference children of r increment their values by the value stored at l. In Figure 6.2, the elements of A that should increment their values are highlighted in (b) with the same colors as the values used to increment them in (a). In the end, A will contain the array locations to store particles to P_{out}. The root node of T gives the total number of elements that P_{out} should have to write those values. Note that, in this algorithm, one separate thread can be assigned for each element of A, with a synchronization stop between iterations.

6.4 Shading by Tangent Vector Advection

Although there exist many techniques for computing normal vectors from point-based surfaces [Hoppe et al., 1992; Zwicker et al., 2002; Schafhitzel et al., 2007], the algorithm presented in this chapter keeps in the data structure only the current and at most also the previous front during integration and rendering. Therefore, the normal vectors can be locally computed by the cross product between the surface tangent vector at each particle position and the velocity vector at that point. For simplicity, the tangent vectors could be locally extracted as the tangent of the front at each particle's position. This method, however, would be susceptible to front deformation (see Figure 6.3a).

Here, the tangent vectors $\mathbf{E} = \{\mathbf{e}_0, \mathbf{e}_1, \dots, \mathbf{e}_{N-1}\}$ are computed at each particle sample position $\mathbf{p}_i$ in P by advecting the tangent vector from the seed curve and keeping it perpendicular to the velocity field at the particle's position. This approach starts by extracting the tangents of the seed curve, which is in the implementation the vector with direction equal to the difference between the next and the previous particles' samples position $\mathbf{p}_{i+1} - \mathbf{p}_{i-1}$. This vector is then adjusted, such that it remains perpendicular to the velocity field and with length equal to ϵ. ϵ is in practice an offset parameter and is set to half δ_{min} or δ'_{min} in physical space, accordingly. At each integration step of this technique, the tangent vectors are advected by also integrating a tangent particle at position $\mathbf{h}_i = \mathbf{p}_i + \mathbf{e}_i$. The tangent vector at the next front is hence selected by adjusting $\mathbf{h}_i - \mathbf{p}_i$ such that it again remains perpendicular to the flow field and with length equal to ϵ. This procedure is performed for each integration step and the normal direction is hence obtained as the cross product $\mathbf{e}_i \times \mathbf{u}(\mathbf{p}_i)$. Figure 6.3 compares the result of shading the surface without and with tangent vector advection in (a) and (b), respectively. This example shows that applying tangent vector advection provides more consistent normals along the surface, leading to a better shading. However, some small artifacts may arise, due to discretization, as this approach does not offer full control over the relation between tangent vectors and the actual tangent of the computed streamsurface.

6.5 Incorporating Texture Advection

Texture-based flow visualization, especially LIC, are important techniques to reveal the intrinsic structure of streamsurfaces. Many existing methods for texture-based visualization on surfaces [Wijk, 2003; Laramee et al., 2003; Weiskopf and Ertl, 2004] are suitable candidates to be combined with this approach. In the experiments, LIC was applied with a simplified implementation of the approach proposed by Weiskopf and Ertl [2004] to demonstrate this feature with the technique. For this, two geometry buffers are attached that, for each image fragment, store vertex positions and normals, both represented in physical space. A 3D noise texture is then replicated in physical space for sampling. The geometry buffers and the noise texture are then used for line integral

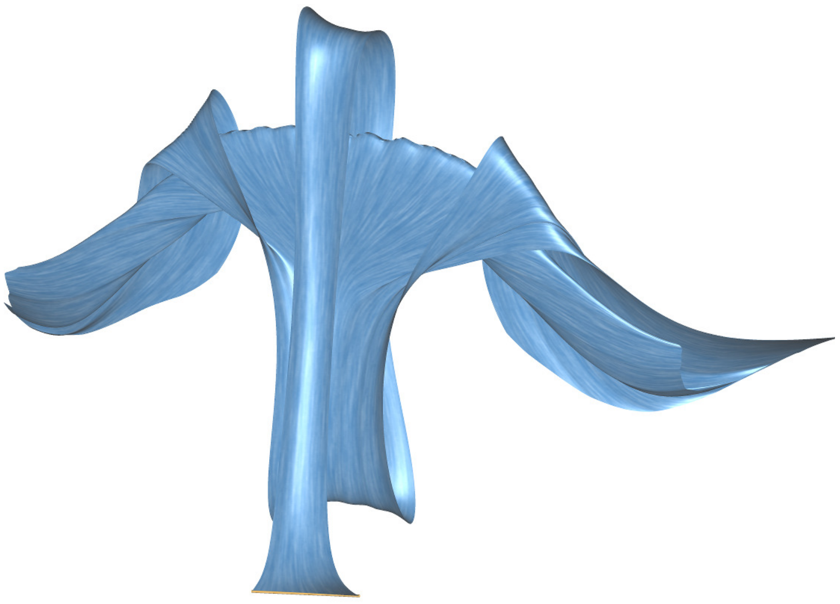


Figure 6.4 — Buoyant Flow I dataset visualized with an image-based streamsurface. The short seed curve (yellow) defines this complex and large streamsurface (blue) because of strong flow divergence.

convolution along the flow field and rendering the final pixel colors, which is further combined with the backbuffer.

6.6 Results

This technique was implemented with CUDA 5.5 and interoperability with OpenGL and OpenGL Shading Language (GLSL). Line integration, front resampling, LIC, and shading are computed with CUDA. The geometry buffer is built with GLSL by rendering the OpenGL primitives *GL_POINTS*, *GL_LINES*, and *GL_LINE_STRIP* for rendering, with the particle-based, streamline-based, and front-based approaches, respectively. The experiments were performed on an Intel Core i5 CPU with 4×3.40 GHz and 16 GiB of RAM, and an NVidia GeForce GTX 770 GPU with 4 GiB.

The technique is demonstrated for the visualization of two buoyant flow CFD datasets. Both datasets are discretized on structured grids with a resolution of $61 \times 31 \times 61$ nodes. For the streamline integration, the RK4 method is employed with step size equal to $\frac{1}{5}$ of the cell size. Figures 2.2b, 6.4, and 6.5 show the respective results. The streamline-based visualization approach was used to generate these images and, for the visualization of intrinsic flow structure, LIC

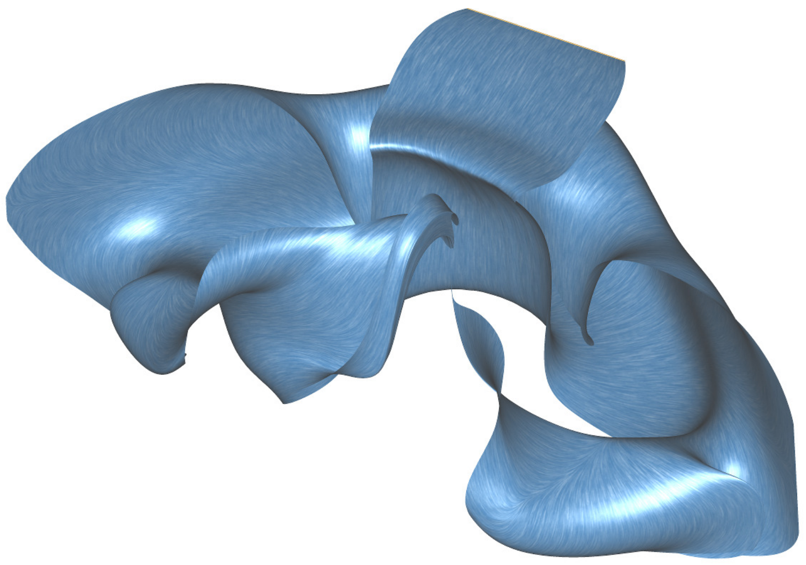


Figure 6.5 — Buoyant Flow II dataset visualized with an image-based streamsurface. The seed curve (yellow) at the top leads to a wide streamsurface (blue) that covers a large region of the whole dataset.

is incorporated. In Figure 2.2b, the streamsurface reveals the flow behavior in the vicinity of a region with vortical flow, 150 time steps after the seed curve (yellow). Note that this approach provides consistent visualization of the surface with coherent shading, though it renders only lines instead of triangles. The results are shown for more complex streamsurfaces in Figures 6.4 and 6.5 to demonstrate the technique's robustness. The streamsurface in Figure 6.4 starts with a short seed curve that turns out to result in a very long front after 375 time steps due to strong flow divergence. Figure 6.5 shows a very wide streamsurface that after 412 time steps covers an area about half of the whole domain.

Table 6.1 gives some timings of this technique for the visualization of the streamsurfaces in the figures with respect to an output image with resolution of 900 × 900 pixels. Note that the time taken by each stage of this approach is somewhat balanced. Due to the GPU-based implementation, the total computation time is hence strongly dependent on the number of integration steps employed (Section 6.7).

Table 6.2 compares the performance of the algorithm for front resampling (Section 6.3) with a single-threaded implementation. The timings for the single-threaded implementation consist of computing the arrays regarding the first step of the fast algorithm. The timings for the second step of the front resample

Table 6.1 — The streamline-based technique's timings for line integration, front resampling, surface rendering, LIC, and total time.

Surface	Time[s]				
	Integration	**Resample**	**Render**	**LIC**	**Total**
Figure 2.2b	0.0409	0.0360	0.0217	0.0284	0.1316
Figure 6.4	0.1034	0.0868	0.0812	0.0582	0.3475
Figure 6.5	0.1264	0.1022	0.1235	0.0970	0.4633

approach are given in the last column. The fast front resampling approach reduces the complexity from the order of $O(n)$ in single-threaded systems to $O(\log(n))$ on parallel systems. Note that, due to the performance gain in the overall front resample time, the fast front-line refinement might be required in order to achieve interactive frame rates.

Table 6.2 — Times to compute the input arrays with a single thread and on the GPU, and the time to perform fast resampling on the GPU after the computation of the input data structure.

Surface	Time[s]		
	Sing.	**GPU**	**Resample**
Figure 2.2b	0.0415	0.0134	0.0225
Figure 6.4	0.1200	0.0324	0.0544
Figure 6.5	0.2172	0.0412	0.0610

Table 6.3 compares the timings of the three image-based approaches presented in Section 6.2.3. Note that the streamline-based approach performs much faster than the other two, and is hence the technique that is recommended for practical use. This technique is fast, and as such, it is the first to offer interactive streamsurface computation with full image-space resolution without user parametrization.

Table 6.3 — Performance comparison for the image-based streamsurfaces with particle-based, front-based, and streamline-based rendering.

Surface	Time[s]		
	Particles	**Fronts**	**Streamlines**
Figure 2.2b	0.8946	0.8461	0.1316
Figure 6.4	1.9919	1.8905	0.3475
Figure 6.5	1.3932	1.2670	0.4633

All timings in Tables 6.1, 6.2, and 6.3 were measured as the mean time of rendering 100 frames with output image resolution of 900 × 900 pixels.

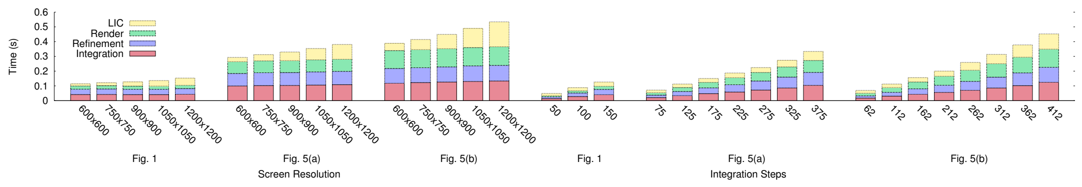


Figure 6.6 — Bar chart graph *Time* × *Resolution* and *Time* × *Steps*.

6.7 Discussion

Although, in the present literature, a comparative study about accuracy and performance of existing streamsurfaces extraction techniques was not found, this section remains on the systematic analysis of the algorithm compared to insights from previous works. Note that such a study would be very complex to design due to the diversity of flows and streamsurfaces, and that previous works also provide their evaluations in such a manner.

The performance of the streamline-based approach is discussed by analyzing separately each step of the algorithm. Employing one integration step to the front in parallel (e.g., on the GPU) can be achieved in constant time. Front resampling can be performed in logarithmic time due to the fast front resampling approach presented in Section 6.3. Thus, the overall performance of this algorithm should be impacted linearly by the number of integration steps, because the method does not interfere with the integration, and logarithmically by the number of front samples, which depends on the screen resolution. The relation between screen resolution and front sampling is empiric as it consists of sampling curves (1D data in 3D space) on the screen (2D space). Note that this algorithm renders lines at each integration step, which is optimized by current graphics cards, and as the other steps of the method output the vertices and normals as arrays directly on the GPU's memory and yet in the correct order for rendering, this step should also tend to be constant in time. More details on the performance of the rendering step should be found in the specifications of the graphics library and the graphics card. Here, an analysis of the impact by dataset complexity is not presented, as it is implicit on the number of integration steps, i.e., it indirectly impacts the technique's performance by impacting the required number of integration steps to be employed.

The stacked bar chart in Figure 6.6 shows the performance behavior of this technique with increase of the screen resolution (left) and number of integration steps (right). It can be observed that even though the screen resolution increases from 600 × 600 to four times more pixels in 1200 × 1200, the impact to performance is minimal. In fact, the overall performance is mainly impacted by the texture advection (yellow) than by this algorithm for computing streamsurfaces (red, blue, and green). The number of integration steps, on the other hand, impacts directly on performance. It can be noted that in the three clusters in Figure 6.6 (right), the regular increase of 50 steps of integration impacts evenly on the time spent with every step of the algorithm, including LIC.

Chapter 7

Conclusion

This thesis presented contributions to vector field visualization that range from feature extraction, integration-based visualization, vector field topology, and solar dynamics visualization. The major contributions of this thesis are to the visualization of vector field topology in bounded domains.

Bifurcation Lines

Local extraction of bifurcation lines was presented (Chapter 3) by altering two existing techniques for vortex core line extraction, formulated by means of the parallel vectors operator. The advantages and drawbacks of the two methods were investigated and, because one of them provided more accurate results in general but failed more often, a refinement of solutions of the parallel vectors operator was introduced. To obtain the 2D manifolds of bifurcation lines, similar to those of saddle-type critical points and saddle-type periodic orbits, a respective seeding strategy that avoids unnecessary gaps in the manifolds was developed. Moreover, the concept of bifurcation lines was related to saddle connectors and saddle-type periodic orbits and it was found that each of these constructs has to contain at least a part of a bifurcation line. As a consequence, the bifurcation line extraction technique provides a local approach to the extraction of (parts of) saddle-type periodic orbits and (parts of) saddle connectors. In the case of periodic orbits, this approach works also in configurations where the velocity along the orbit is low compared to the hyperbolicity of the orbit, cases where traditional approaches based on streamline integration tend to fail.

The approach for extracting the 2D manifolds of bifurcation lines was shown to even extract periodic orbits (and saddle connectors) that are not hyperbolic everywhere along these constructs. It has also been shown that bifurcation lines and their 2D manifolds can complete traditional vector field topology.

Solar Dynamics

Visualization techniques from flow visualization and volume rendering were successfully applied to solar dynamics data (Chapter 4). While already the straightforward application of existing techniques from these fields proved valuable for solar physics research, the domain experts identified high potential in their combination. In particular, they expect that the presented space-time visualization of the magnetogram structure and 3D LIC may become widely-used tools in this field. Although they were fascinated about the new virtual coronal loop visualization technique, it is the technique that is hardest to interpret and validate in terms of coronal plasma dynamics. Nevertheless, since the resulting loops strictly follow the magnetic field, their utility for the visualization of the magnetic field as a method that imitates the self-illustrating phenomenon of coronal loops, is out of question. The domain experts expect an enrichment in research both on the astrophysics and visualization side due to this work.

In contrast to previous visualization tools like JHelioviewer [jhe], which concentrate on 2D display, or IDL-based software packages like SSW [ssw], which has limited visualization capabilities, the approaches presented in this thesis offer an integrated system for space and time navigation in a three-dimensional context, and interactive exploration of local magnetic field structure.

Escape Maps

This thesis presented in Chapter 5 a technique for topologically correct and robust extraction of escape maps, which show if a streamline emanating from a given position on the map in forward or reverse direction leaves an adjacent, arbitrarily-shaped simply-connected subregion of the domain. The concept of escape maps was inspired from astrophysics, where it is equivalent to the problem of extracting coronal holes on the Sun's photosphere. In contrast to the traditional approach that discretizes the map and therefore suffers from discretization issues, the approach presented in this thesis builds on isocline surfaces of the vector field to obtain a seeding tailored at the topologically correct extraction of escape maps. This approach provides several benefits. First and most important, it is able to extract very narrow map structures, in contrast to the traditional approach that suffers severely from aliasing. This becomes particularly obvious in case of the extraction of coronal hole corridors, which are missed by traditional approaches and are subject to current research in astrophysics. Second, it performs field line integration in these cases in the more stable direction toward the map boundary, avoiding prohibitive accumulation of integration error. This technique could be particularly helpful in astrophysics research on coronal hole corridors, e.g., regarding the conjecture by Antiochos et al. [2007].

Image-Based Streamsurfaces

This thesis also presents an image-based technique for the computation of streamsurfaces (Chapter 6). This technique computes and renders only lines (or points) in order to avoid the expensive triangulation involved in most of the previous techniques. The technique is very simple to implement because it does not adopt complex criteria for dealing with flow divergence and convergence. It is also independent of extra user-defined parametrization to avoid perceivable discretization artifacts (gaps) on the visualizations. A GPU optimization was demonstrated to handle front resampling during streamsurface integration, which reduces the complexity of the front resampling from $O(n)$ to $O(\log(n))$ by adopting a divide and conquer strategy for the GPU. Three different approaches were analyzed for the image-based streamsurfaces (i.e., particle-based, streamline-based, and front-based) and it was concluded that due to the performance gains over the other two, the streamline-based approach is most useful for practical visualization.

Future Work

Some ideas for possible future work have arised in this work. For example, a further study could apply the refinement of the solutions of the parallel vectors operator, presented in Section 3.1.2, for a more accurate extraction of vortex core lines and ridge lines. Furthermore, the investigation or development of more sophisticated refinement approaches could also be subject of such future work. In solar physics visualization, the following topics lend themselves as future work. The space-time investigation approach is limited to phenomena on the solar surface. In the future, one could investigate dedicated means for studying 3D phenomena, such as coronal loops or faculae and the spectral power distribution including all AIA bands, over time. The potential-field source-surface (PFSS) method is still too slow for interactive rates with reasonable resolution. Hence, one could incorporate suitable preconditioners as well as parallel sparse linear solvers for GPU clusters. Vector magnetograms could significantly improve the local magnetic field extrapolation using PFSS or could also be used for direct visualization. Hopefully, vector magnetograms will become available from the JSOC database in the near future. It could also be tested if the solar region summary (SRS), which is manually compiled by domain experts, could be complemented by automated data mining systems, similar to the Heliophysics Event Knowledgebase (HEK) Hurlburt et al. [2012] system. The concept of escape maps is closely related to vector field topology on bounded domains. Nevertheless, further research is necessary to clearly determine possible links and properties between these two concepts, and if escape maps could provide a basis for such a vector field topology. Future research could investigate the connection between isocline surfaces and topological features. Moreover, a comparative study of the accuracy of the image-based streamsurfaces with those from previous work could be performed. The accuracy of the image-based approach is mainly limited by the resolution of the output image. Hence, visualization on ultra-high resolution displays is expected to provide highly accurate results. Finally, further investigation of the image-based integration can be performed to study the behaviour of streamlines in the vicinity of very small structures like critical points, as this approach automatically performs streamline adaptation during navigation.

APPENDIX A

Coronal Magnetic Field Extrapolation on GPU

While the magnetic field can readily be measured spectroscopically in the solar photosphere, this is unfortunately not feasible in the solar corona due to its much lower density. For this reason, extrapolation methods are needed to reconstruct the coronal magnetic field based on photospheric data. The simplest one—the Potential-Field Source-Surface (PFSS) model—was developed by Altschuler and Newkirk [1969] and Schatten et al. [1969]. It assumes that the coronal magnetic field is current-free,

$$\nabla \times \mathbf{B} = \mathbf{0}, \tag{A.1}$$

and thus can be represented by the gradient of a scalar potential,

$$\mathbf{B} = \nabla \psi. \tag{A.2}$$

Together with the divergence free condition of a magnetic field,

$$\nabla \cdot \mathbf{B} = 0, \tag{A.3}$$

the scalar potential has to fulfill the Laplacian equation

$$\nabla^2 \psi = 0. \tag{A.4}$$

For the solar corona, the Laplacian is transformed into standard spherical coordinates

$$x = r \sin \vartheta \cos \varphi, \quad y = r \sin \vartheta \sin \varphi, \quad z = r \cos \vartheta \tag{A.5}$$

with boundary conditions only at constant radii. At the photosphere, $r = R_\odot$, there is the Neumann boundary condition

$$\left.\frac{\partial\psi}{\partial r}\right|_{r=R_\odot} = M(\cos\vartheta, \varphi). \tag{A.6}$$

with $\vartheta = (0, \pi)$, $\varphi = [0, 2\pi)$, and the values from the corresponding magnetogram, $M(\cos\vartheta, \varphi)$. The outer boundary at $r = R_w = 2.5R_\odot$ is given by $\psi_r = 0$, where the field is assumed to be purely radial. The spherical magnetic field components then follow from

$$B_r = -\frac{\partial\psi}{\partial r}, \quad B_\vartheta = -\frac{1}{r}\frac{\partial\psi}{\partial\vartheta}, \quad B_\varphi = -\frac{1}{r\sin\vartheta}\frac{\partial\psi}{\partial\varphi}. \tag{A.7}$$

The relation between these spherical components and the corresponding Cartesian magnetic field vector

$$\mathbf{B} = B_r\mathbf{e}_r + B_\vartheta\mathbf{e}_\vartheta + B_\varphi\mathbf{e}_\varphi \tag{A.8}$$

is given by the spherical basis vectors

$$\mathbf{e}_r = (\sin\vartheta\cos\varphi, \sin\vartheta\sin\varphi, \cos\vartheta)^T, \tag{A.9a}$$

$$\mathbf{e}_\vartheta = (\cos\vartheta\cos\varphi, \cos\vartheta\sin\varphi, -\sin\vartheta)^T, \tag{A.9b}$$

$$\mathbf{e}_\varphi = (-\sin\varphi, \cos\varphi, 0)^T. \tag{A.9c}$$

In this work, however, the spherical LCEA coordinates (r, μ, φ) is used where $\mu = \sin\theta = \cos\vartheta$ with latitude θ instead of colatitude ϑ as used in traditional spherical coordinates. The LCEA coordinates are related to Cartesian coordinates via

$$\mathbf{\Gamma} : x = r\sqrt{1-\mu^2}\cos\varphi, \quad y = r\sqrt{1-\mu^2}\sin\varphi, \quad z = r\mu, \tag{A.10a}$$

$$\mathbf{\Gamma}^{-1} : \varphi = \arctan\frac{y}{x}, \quad \mu = \frac{z}{\sqrt{x^2+y^2+z^2}}, \quad r = \sqrt{x^2+y^2+z^2}. \tag{A.10b}$$

The magnetic field vector $\mathbf{B}$, Equation A.8, is related to the gradient of the potential ψ with respect to LCEA coordinates in the following way:

$$\mathbf{B} = \begin{pmatrix} \sqrt{1-\mu^2}\cos\varphi & \frac{\mu\sqrt{1-\mu^2}}{r}\cos\varphi & -\frac{\sin\varphi}{r\sqrt{1-\mu^2}} \\ \sqrt{1-\mu^2}\sin\varphi & \frac{\mu\sqrt{1-\mu^2}}{r}\cos\varphi & -\frac{\sin\varphi}{r\sqrt{1-\mu^2}} \\ \mu & -\frac{1-\mu^2}{r} & 0 \end{pmatrix} \begin{pmatrix} \frac{\partial\psi}{\partial r} \\ \frac{\partial\psi}{\partial\mu} \\ \frac{\partial\psi}{\partial\varphi} \end{pmatrix}, \tag{A.11}$$

In this work, *computational space* (c-space) spanned by the LCEA coordinates is distinguished from *physical space* (p-space) corresponding to the Cartesian coordinates.

A.1 Spherical Harmonics

The solution to the Laplacian equation (Equation A.4) in spherical coordinates is usually expanded into spherical harmonics. Following Altschuler and Newkirk [1969], the magnetic potential ψ in the domain $R_\odot \leq r \leq R_w = 2.5R_\odot$ reads

$$\psi(r,\mu,\varphi) = R_\odot \sum_{n=1}^{\infty} \sum_{m=0}^{n} \left\{ \rho_n \left[g_n^m \cos(m\varphi) + h_n^m \sin(m\varphi) \right] P_n^m(\mu) \right\} \tag{A.12}$$

with

$$\rho_n = \left[\left(\frac{r}{R_\odot}\right)^n - a_n \left(\frac{R_\odot}{r}\right)^{n+1} \right] \frac{1}{a_n - 1}, \quad a_n = \left(\frac{R_w}{R_\odot}\right)^{2n+1}, \tag{A.13}$$

and Schmidt-normalized associate Legendre polynomials $P_n^m(\mu)$,

$$\int_{\mu=-1}^{1} \int_{\varphi=0}^{2\pi} P_n^m(\mu) \begin{Bmatrix} \cos(m\varphi) \\ \sin(m\varphi) \end{Bmatrix} P_{n'}^{m'}(\mu) \begin{Bmatrix} \cos(m'\varphi) \\ \sin(m'\varphi) \end{Bmatrix} d\mu \, d\varphi$$
$$= \frac{4\pi}{2n+1} \delta_{nn'} \delta_{mm'}. \tag{A.14}$$

Please note that μ is used instead of $\cos\vartheta$. The spherical harmonic coefficients g_n^m and h_n^m up to $n_{max} = 40$ are provided by GONG [gon].

A.2 PFSS

As pointed out by Tóth et al. [2011], the traditional spherical harmonics decomposition works reasonably well when the order of spherical harmonics is limited to be small relative to the resolution of the magnetogram. However, around sharp features, ringing artifacts become inevitable. Furthermore, spherical harmonics are global functions and their amplitudes depend on all of the magnetogram data. Since the basis of the spherical harmonics are the synoptic magnetograms, the global magnetic field is influenced by data that might have already changed significantly.

The finite difference method for PFSS has the advantage to be applicable also for local domains. In this work, after selecting a domain on the visible side of the Sun, a regular curvilinear grid is defined bounded by either $\mu = \text{const}$, $\varphi = \text{const}$, or $r = \text{const}$. The lower boundary condition, $r = R_\odot = 1$, is defined by a line-of-sight HMI magnetogram which is resampled for this grid. The other five faces follow from the spherical harmonics model using the coefficients by GONG.

The finite difference operator of the Laplacian equation $\nabla^2\psi = 0$ in spherical LCEA coordinates (r, μ, φ) reads

$$\begin{aligned}
\nabla^2\psi_{i,j,k} = {} & \frac{\psi_{i+1,j,k} - \psi_{i-1,j,k}}{r_i\Delta r} + \frac{\psi_{i+1,j,k} - 2\psi_{i,j,k} + \psi_{i-1,j,k}}{\Delta r^2} \\
& - \frac{\mu_j}{r_i^2}\frac{\psi_{i,j+1,k} - \psi_{i,j-1,k}}{\Delta\mu} + \frac{1-\mu_j^2}{r_i^2}\frac{\psi_{i,j+1,k} - 2\psi_{i,j,k} + \psi_{i,j-1,k}}{\Delta\mu^2} \\
& + \frac{1}{r_i^2(1-\mu_j^2)}\frac{\psi_{i,j,k+1} - 2\psi_{i,j,k} + \psi_{i,j,k-1}}{\Delta\varphi^2}.
\end{aligned}$$

The boundary condition, Equation A.6, is realized by setting

$$\psi_{0,j,k} = \psi_{1,j,k} - \Delta r M_{j,k}. \tag{A.15}$$

Linearization of the scalar field

$$\psi_{i,j,k} \mapsto \Psi[(iN_{th} + j)N_{ph} + k] \tag{A.16}$$

yields a sparse linear system $A\Psi = b$ which is solved by the Krylov-type iterative method BiCGSTAB from the Cusp-library Bell and Garland [2012]. As initial guess, we set $\Psi = 0$ inside the computational domain.

Bibliography

CitcomS. http://geodynamics.org/cig/software/citcoms. 93

National solar observatory, global oscillation network group (GONG). http://gong.nso.edu/. 51, 117

JHelioviewer. http://jhelioviewer.org/. 48, 50, 112

Predictive science modeling support for helioseismic and magnetic imager solar dynamics observatory. http://www.predsci.com/hmi/home.php. 44, 52, 62, 63, 88

SolarSoftWare (SSW) is an interactive data language (IDL) based software library. www.lmsal.com/solarsoft. 50, 112

M. D. Altschuler and G. Newkirk. Magnetic fields and the structure of the solar corona. *Solar Physics*, 9:131–149, 1969. 50, 53, 115, 117

S. K. Antiochos, C. R. DeVore, J. T. Karpen, and Z. Mikić. Structure and dynamics of the Sun's open magnetic field. *Astrophysical Journal*, 671(1):936–946, 2007. 91, 113

D. Asimov. Notes on the topology of vector fields and flows. Technical Report RNR-93-003, NASA Ames Research Center, 1993. 1, 4, 19, 21

S. Bachthaler, F. Sadlo, R. Weeber, S. Kantorovich, C. Holm, and D. Weiskopf. Magnetic flux topology of 2D point dipoles. *Computer Graphics Forum*, 31(3): 955–964, 2012. 21, 49

N. Bell and M. Garland. Cusp: Generic parallel algorithms for sparse matrix and graph computations, 2012. [Online]. Available: cusp-library.googlecode.com. 54, 66, 118

B. Cabral and L. C. Leedom. Imaging vector fields using line integral convolution. In *Proceedings of SIGGRAPH Conference on Computer Graphics and Interactive Techniques*, pages 263–270, 1993. 2, 14, 96

D. Cai, B. Lembege, and K. Nishikawa. Visualizing magnetospheric vector field topology. In *Proceedings of ISSS-7*, pages 26–31, 2005. 50

S. R. Cranmer. Coronal holes. *Living Reviews in Solar Physics*, 6(3), 2009. 50, 61, 66

W. de Leeuw and R. van Liere. Comparing LIC and spot noise. In *Proceedings of IEEE Visualization*, pages 359–365, 1998. 15

W. C. de Leeuw and J. J. van Wijk. A probe for local flow field visualization. In *Proceedings of IEEE Visualization*, pages 39–45, 1993. 8

M. DeRosa. Interactive data language (IDL) software for analyzing solar magnetic fields. www.lmsal.com/~derosa/pfsspack. 50

L. K. Forssell. Visualizing flow over curvilinear grid surfaces using line integral convolution. In *Proceedings of IEEE Visualization*, pages 240–247, 1994. 58

R. Fuchs, R. Peikert, H. Hauser, F. Sadlo, and P. Muigg. Parallel vectors criteria for unsteady flow vortices. *IEEE Transactions on Visualization and Computer Graphics*, 14(3):615–626, 2008. 24

C. Garth, X. Tricoche, T. Salzbrunn, T. Bobach, and G. Scheuermann. Surface techniques for vortex visualization. In *Proceedings of EG/VGTC Conference on Visualization*, pages 155–164, 2004. 95

C. Garth, H. Krishnan, X. Tricoche, T. Tricoche, and K. I. Joy. Generation of accurate integral surfaces in time-dependent vector fields. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1404–1411, 2008. 5, 13, 95

J. A. Gilbert, T. H. Zurbuchen, and L. A. Fisk. A new technique for mapping open magnetic flux from the solar surface into the heliosphere. *Astrophysical Journal*, 663:583–591, 2007. 66

A. Globus, C. Levit, and T. Lasinski. A tool for visualizing the topology of three-dimensional vector fields. In *Proceedings of IEEE Visualization*, pages 33–40, 408, 1991. 1, 8, 15

T. Günther, K. Bürger, R. Westermann, and H. Theisel. A view-dependent and inter-frame coherent visualization of integral lines using screen contribution. In *Proceedings of Workshop on Vision, Modeling, and Visualization*, pages 215–222, 2011. 11

G. Haller. Finding finite-time invariant manifolds in two-dimensional velocity fields. *Chaos*, 10(1):99–108, 2000. 29

H. Hauser and E. Gröller. Thorough insights by enhanced visualization of flow topology. In *Proceedings of International Symposium on Flow Visualization*, 2000. 15

J. L. Helman and L. Hesselink. Representation and display of vector field topology in fluid flow data sets. *Computer*, 22(8):27–36, 1989. 1, 15

J. L. Helman and L. Hesselink. Visualizing vector field topology in fluid flows. *IEEE Computer Graphics and Applications*, 11(3):36–46, 1991. 2, 15, 25

M. Hlawatsch, F. Sadlo, H. Jang, and D. Weiskopf. Pathline glyphs. *Computer Graphics Forum*, 33(2):497–506, 2014. 8

H. Hoppe, T. DeRose, T. Duchamp, J. Mcdonald, and W. Stuetzle. Surface reconstruction from unorganized points. In *Proceedings of SIGGRAPH Conference on Computer Graphics and Interactive Techniques*, pages 71–78, 1992. 104

J. P. M. Hultquist. Constructing stream surfaces in steady 3D vector fields. In *Proceedings of IEEE Visualization*, pages 171–178, 1992. 1, 5, 12, 82, 95, 96

N. Hurlburt et al. Heliophysics event knowledgebase for the solar dynamics observatory (SDO) and beyond. *Solar Physics*, 275:67–78, 2012. 114

V. Interrante and C. Grosch. Strategies for effectively visualizing 3D flow with volume LIC. In *Proceedings of IEEE Visualization*, pages 421–424, 1997. 14

V. Interrante and C. Grosch. Visualizing 3D flow. *IEEE Computer Graphics and Applications*, 18(4):49–53, 1998. 14

J. Jeong and F. Hussain. On the identification of a vortex. *Journal of Fluid Mechanics*, 285(69):69–94, 1995. 25

B. Jobard and W. Lefer. Creating evenly-spaced streamlines of arbitrary density. pages 43–56. 1997. 1, 10

S. W. Kahler and H. S. Hudson. Boundary structures and changes in long-lived coronal holes. *Astrophysical Journal*, 574(1):467–476, 2002. 91

J. Kasten, J. Reininghaus, W. Reich, and G. Scheuermann. Toward the extraction of saddle periodic orbits. In *Topological Methods in Data Analysis and Visualization III*, Mathematics and Visualization, pages 55–69. 2014. 18

D. N. Kenwright, C. Henze, and C. Levit. Feature extraction of separation and attachment lines. *IEEE Transactions on Visualization and Computer Graphics*, 5 (2):135–144, 1999. 2, 4, 25, 27

T. Klein and T. Ertl. Illustrating magnetic field lines using a discrete particle model. In *Proceedings of Workshop on Vision, Modeling, and Visualization*, pages 387–394, 2004. 49

H. Krishnan, C. Garth, and K. Joy. Time and streak surfaces for flow visualization in large time-varying data sets. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1267–1274, 2009. 5, 13, 95

L. Krista and P. Gallagher. Automated coronal hole detection using local intensity thresholding techniques. *Solar Physics*, 256:87–100, 2009. 50

D. H. Laidlaw, R. M. Kirby, C. D. Jackson, J. S. Davidson, T. S. Miller, M. da Silva, W. H. Warren, and M. J. Tarr. Comparing 2D vector field visualization methods: A user study. *IEEE Transactions on Visualization and Computer Graphics*, 11(1):59–70, 2005. 8

R. S. Laramee, B. Jobard, and H. Hauser. Image space based visualization of unsteady flow on surfaces. In *Proceedings of IEEE Visualization*, pages 131–138, 2003. 14, 105

R. S. Laramee, H. Hauser, H. Doleisch, B. Vrolijk, F. H. Post, and D. Weiskopf. The state of the art in flow visualization: Dense and texture-based techniques. *Computer Graphics Forum*, 23(2):203–222, 2004. 15

Y. Levy, D. Degani, and A. Seginer. Graphical visualization of vortical flows by means of helicity. *AIAA*, 28(8):1347–1352, 1990. 24

L. Li and H.-W. Shen. Image-based streamline generation and rendering. *IEEE Transactions on Visualization and Computer Graphics*, 13:630–640, 2007. 11

L. Li, H.-H. Hsieh, and H.-W. Shen. Illustrative streamline placement and visualization. In *Proceedings of IEEE Pacific Visualization Symposium*, pages 79–86, 2008. 11

W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In *Proceedings of SIGGRAPH Conference on Computer Graphics and Interactive Techniques*, pages 163–169, 1987. 13, 83

G. Machado, F. Sadlo, and T. Ertl. Image-based streamsurfaces. In *Proceedings of Conference on Graphics, Patterns and Images (SIBGRAPI)*, pages 343–350, 2014a. 4

G. Machado, F. Sadlo, T. Muller, and T. Ertl. Escape maps. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2604–2613, 2014b. 4

G. M. Machado, F. Sadlo, T. Müller, D. Müller, and T. Ertl. Visualizing solar dynamics data. In *Proceedings of Workshop on Vision, Modeling, and Visualization*, pages 95–102, 2012. 4

G. M. Machado, F. Sadlo, and T. Ertl. Local extraction of bifurcation lines. In *Proceedings of Workshop on Vision, Modeling, and Visualization*, pages 17–24, 2013. 4, 80

D. Mackay and A. Yeates. The Sun's global photospheric and coronal magnetic fields: Observations and models. *Living Reviews in Solar Physics*, 9(6), 2012. 52

O. Mallo, R. Peikert, C. Sigg, and F. Sadlo. Illuminated lines revisited. In *Proceedings of IEEE Visualization*, pages 19–26, 2005. 11

J. Martinez Esturo, M. Schulze, C. Rössl, and H. Theisel. Poisson-based tools for flow visualization. In *Proceedings of IEEE Pacific Visualization Symposium*, pages 241–248, 2013. 36

T. McLoughlin, R. S. Laramee, and E. Zhang. Easy integral surfaces: A fast, quad-based stream and path surface algorithm. In *Proceedings of Computer Graphics International Conference*, pages 73–82, 2009. 13

T. McLoughlin, R. S. Laramee, R. Peikert, F. H. Post, and M. Chen. Over two decades of integration-based, geometric flow visualization. *Computer Graphics Forum*, 29(6):1807–1829, 2010. 1, 14

A. Mebarki, P. Alliez, and O. Devillers. Farthest point seeding for efficient placement of streamlines. In *Proceedings of IEEE Visualization*, page 61, 2005. 11

D. Müller et al. JHelioviewer: Visualizing large sets of solar images using JPEG 2000. *Computing in Science & Engineering*, 11(5):38–47, 2009. 48, 50

M. Otto and H. Theisel. Vortex analysis in uncertain vector fields. *Computer Graphics Forum*, 31(3pt2):1035–1044, 2012. 15

M. Otto, T. Germer, and H. Theisel. Uncertain topology of 3D vector fields. In *Proceedings of IEEE Pacific Visualization Symposium*, pages 67–74, 2011. 15

R. Peikert and M. Roth. The parallel vectors operator: a vector field visualization primitive. In *Proceedings of IEEE Visualization*, pages 263–270, 1999. 23, 27

R. Peikert and F. Sadlo. Topology-guided visualization of constrained vector fields. In *Topology-Based Methods in Visualization*, pages 21–34. 2007. 15

R. Peikert and F. Sadlo. Height ridge computation and filtering for visualization. In *Proceedings of IEEE Pacific Visualization Symposium*, pages 119–126, 2008. 23

W. D. Pence et al. Definition of the flexible image transport system (FITS), version 3.0. *Astronomy and Astrophysics*, 524:A42, 2010. 51

A. E. Perry and M. S. Chong. A description of eddying motions and flow patterns using critical-point concepts. *Annual Review of Fluid Mechanics*, 19: 125–155, 1987. 1, 2, 15, 27, 28, 32

F. H. Post, B. Vrolijk, H. Hauser, R. S. Laramee, and H. Doleisch. The state of the art in flow visualisation: Feature extraction and tracking. *Computer Graphics Forum*, 22(4):775–792, 2003. 23, 25

C. Rezk-Salama, P. Hastreiter, C. Teitzel, and T. Ertl. Interactive exploration of volume line integral convolution based on 3D-texture mapping. In *Proceedings of IEEE Visualization*, pages 233–528, 1999. 14, 58

P. Riley, J. A. Linker, Z. Mikić, R. Lionello, S. A. Ledvina, and J. G. Luhmann. A comparison between global solar magnetohydrodynamic and potential field source surface model results. *Astrophysical Journal*, 653(2):1510–1516, 2006. 50

M. Roth. *Automatic extraction of vortex core lines and other line-type features for scientific visualization*. PhD thesis, ETH Zurich, No. 13673, 2000. 2, 4, 27, 33

M. Roth and R. Peikert. Flow visualization for turbomachinery design. In *Proceedings of IEEE Visualization*, pages 381–384, 1996. 24

M. Roth and R. Peikert. A higher-order method for finding vortex core lines. In *Proceedings of IEEE Visualization*, pages 143–150, 1998. 2, 4, 25, 27, 28, 32, 34

T. Roudier et al. Families of fragmenting granules and their relation to meso- and supergranular flow fields. *Astronomy & Astrophysics*, 409:299–308, 2003. 62

F. Sadlo. *Computational visualization of physics and topology in unstady flow*. PhD thesis, ETH Zurich, No.19284, 2010. 25

F. Sadlo, R. Peikert, and E. Parkinson. Vorticity based flow analysis and visualization for pelton turbine design optimization. In *Proceedings of IEEE Visualization*, pages 179–186, 2004. 49

J. Sahner, T. Weinkauf, and H.-C. Hege. Galilean invariant extraction and iconic representation of vortex core lines. In *Proceedings of Eurographics/IEEE VGTC Symposium on Visualization*, pages 151–160, 2005. 25

A. R. Sanderson, G. Chen, X. Tricoche, D. Pugmire, S. Kruger, and J. Breslau. Analysis of recurrent patterns in toroidal magnetic fields. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1431–1440, 2010. 49

T. Schafhitzel, E. Tejada, D. Weiskopf, and T. Ertl. Point-based stream surfaces and path surfaces. In *Proceedings of Graphics Interface*, pages 289–296, 2007. 5, 13, 96, 103, 104

T. Schafhitzel, J. E. Vollrath, J. P. Gois, D. Weiskopf, A. Castelo, and T. Ertl. Topology-preserving lambda2-based vortex core line detection for flow visualization. *Computer Graphics Forum*, 27(3):1023–1030, 2008. 25

K. Schatten, J. Wilcox, and N. Ness. A model of interplanetary and coronal magnetic fields. *Solar Physics*, 6:442–455, 1969. 50, 53, 61, 115

G. Scheuermann, H. Hagen, H. Krüger, M. Menzel, and A. Rockwood. Visualization of higher order singularities in vector fields. In *Proceedings of IEEE Visualization*, pages 67–74, 1997. 18

G. Scheuermann, B. Hamann, K. I. Joy, and W. Kollmann. Localizing vector field topology. In *Data Visualization: The State of the Art*, pages 19–36. 2003. 2, 3, 22

C. Schrijver et al. Nonlinear force-free modeling of coronal magnetic fields part I: A quantitative comparison of methods. *Solar Physics*, 235:161–190, 2006. 50

M. Schulze, T. Germer, C. Rössl, and H. Theisel. Stream surface parametrization by flow-orthogonal front lines. *Computer Graphics Forum*, 31(5):1725–1734, 2012. 5, 13, 95

H.-W. Shen and D. L. Kao. UFLIC: a line integral convolution algorithm for visualizing unsteady flows. In *Proceedings of IEEE Visualization*, pages 317–322, 1997. 14

B. Spencer, R. S. Laramee, G. Chen, and E. Zhang. Evenly spaced streamlines for surfaces: An image-based approach. *Computer Graphics Forum*, 28(6): 1618–1631, 2009. 11

D. Stalling and H.-C. Hege. Fast and resolution independent line integral convolution. In *Proceedings of SIGGRAPH Conference on Computer Graphics and Interactive Techniques*, pages 249–256, 1995. 14

T. Stöter, T. Weinkauf, H.-P. Seidel, and H. Theisel. Implicit integral surfaces. In *Proceedings of Workshop on Vision, Modeling, and Visualization*, pages 127–134, 2012. 5, 14, 95

D. Sujudi and R. Haimes. Identification of swirling flow in 3D vector fields. In *Proceedings of AIAA Computational Fluid Dynamics Conference*, pages 95–1715, 1995. 2, 4, 24, 25, 27, 28, 32, 34

A. Sundquist. Dynamic line integral convolution for visualizing streamline evolution. *IEEE Transactions on Visualization and Computer Graphics*, 9(3):273–282, 2003. 14, 49

H. Theisel, T. Weinkauf, H.-C. Hege, and H.-P. Seidel. Saddle connectors - an approach to visualizing the topological skeleton of complex 3D vector fields. In G. Turk, J. J. van Wijk, and R. Moorhead, editors, *Proceedings of IEEE Visualization*, pages 225–232, 2003. 4, 8, 21, 22, 95

B. Thomaszewski, A. Gumann, S. Pabst, and W. Strasser. Magnets in motion. *ACM Transactions on Graphics*, 27(5):162:1–162:9, 2008. 49

J. F. Thompson, B. K. Soni, and N. P. Weatherill. *Handbook of Grid Generation*. Taylor & Francis, 1998. 73

W. T. Thompson. Coordinate systems for solar image data. *Astronomy and Astrophysics*, 449:791–803, 2006. 51

V. S. Titov, Z. Mikić, J. A. Linker, R. Lionello, and S. K. Antiochos. Magnetic topology of coronal hole linkages. *Astrophysical Journal*, 731(2):111, 2011. 91

G. Tóth, B. van der Holst, and Z. Huang. Obtaining potential field solutions with spherical harmonics and finite differences. *Astrophysical Journal*, 732(2): 102, 2011. 53, 117

X. Tricoche, G. Scheuermann, and H. Hagen. Higher order singularities in piecewise linear vector fields. In *The Mathematics of Surfaces IX*, pages 99–113. 2000. 18

G. Turk and D. Banks. Image-guided streamline placement. In *Proceedings of SIGGRAPH Conference on Computer Graphics and Interactive Techniques*, pages 453–460, 1996. 1, 10

M. Üffinger, F. Sadlo, and T. Ertl. A time-dependent vector field topology based on streak surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 19(3):379–392, 2013. 15

J. J. van Wijk. Spot noise texture synthesis for data visualization. In *Proceedings of SIGGRAPH Conference on Computer Graphics and Interactive Techniques*, number 4, pages 309–318, 1991. 14

J. J. van Wijk. Implicit stream surfaces. In G. M. Nielson and R. D. Bergeron, editors, *Proceedings of IEEE Visualization*, pages 245–252, 1993. 5, 13, 95

J. Villasenor and A. Vincent. An algorithm for space recognition and time tracking of vorticity tubes in turbulence. *CVGIP: Image Understanding*, 55(1): 27–35, 1992. 24

Y.-M. Wang and N. R. Sheeley, Jr. On potential field models of the solar corona. *Astrophysical Journal*, 392:310–319, 1992. 66

R. Wegenkittl, E. Gröller, and W. Purgathofer. Animating flowfields: Rendering of oriented line integral convolution. In *Proceedings of Computer Animation*, pages 15–21, 1996. 14

T. Weinkauf. *Extraction of topological structures in 2D and 3D vector fields*. PhD thesis, Otto-von-Guericke-Universität Magdeburg, 2008. 18

T. Weinkauf, H. Theisel, H.-C. Hege, and H.-P. Seidel. Boundary switch connectors for topological visualization of complex 3D vector fields. In *Proceedings of Eurographics/IEEE VGTC Symposium on Visualization*, pages 183–192, 2004. 2, 22, 31, 73, 80

T. Weinkauf, H. Theisel, K. Shi, H.-C. Hege, and H.-P. Seidel. Extracting higher order critical points and topological simplification of 3D vector fields. In *Proceedings of IEEE Visualization*, pages 559–566, 2005. 18

T. Weinkauf, J. Sahner, H. Theisel, and H.-C. Hege. Cores of swirling particle motion in unsteady flows. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1759–1766, 2007. 24

D. Weiskopf and T. Ertl. A hybrid physical/device-space approach for spatio-temporally coherent interactive texture advection on curved surfaces. In *Proceedings of Graphics Interface*, pages 263–270, 2004. 5, 14, 105

T. Wiegelmann et al. How to optimize nonlinear force-free coronal magnetic field extrapolations from SDO/HMI vector magnetograms? *arXiv:1202.3601v1*, 2012. 50

J. J. v. Wijk. Image based flow visualization for curved surfaces. In *Proceedings of IEEE Visualization*, pages 123–130, 2003. 14, 105

T. Wischgoll and G. Scheuermann. Locating closed streamlines in 3D vector fields. In *Proceedings of the Eurographics/IEEE VGTC Symposium on Data Visualisation*, pages 227–232, 2002. 18

K. Wu, Z. Liu, S. Zhang, and R. J. M. II. Topology-aware evenly spaced streamline placement. *IEEE Transactions on Visualization and Computer Graphics*, 16(5):791–801, 2010. 11

M. Zöckler, D. Stalling, and H.-C. Hege. Interactive visualization of 3D-vector fields using illuminated stream lines. In *Proceedings of IEEE Visualization*, pages 107–113, 1996. 1, 11

M. Zöckler, D. Stalling, and H.-C. Hege. Parallel line integral convolution. In *Proceedings of Eurographics Workshop on Parallel Graphics and Visualization*, pages 111–128, 1996. 14

M. Zwicker, H. Pfister, J. van Baar, and M. Gross. EWA splatting. *IEEE Transactions on Visualization and Computer Graphics*, 8(3):223–238, Jul 2002. 104