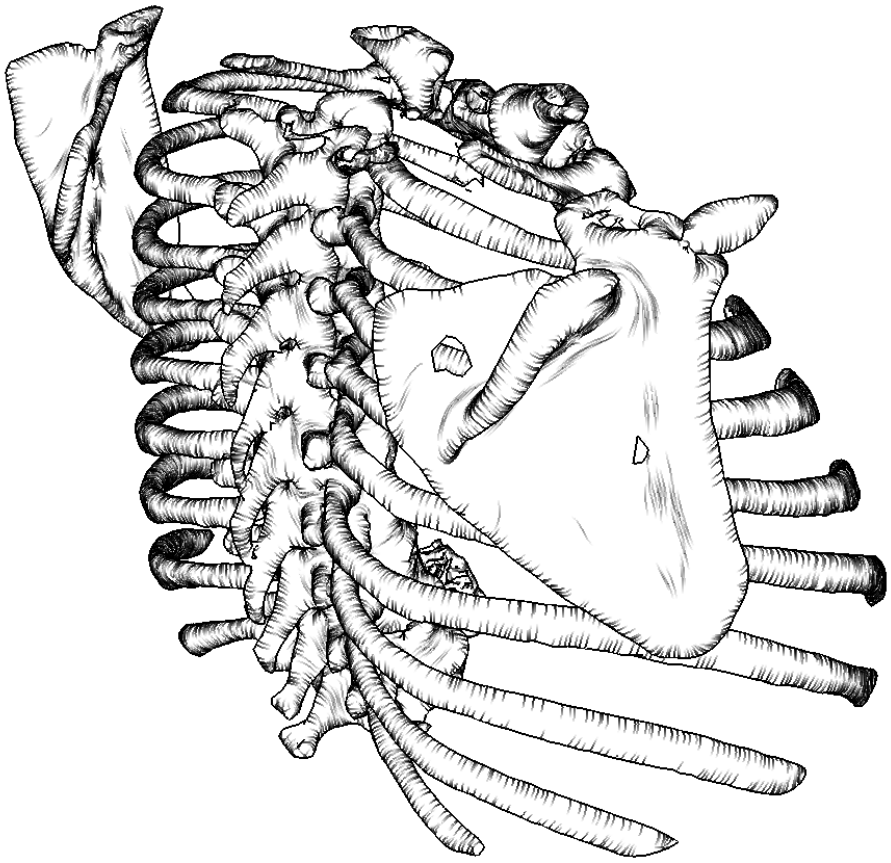
KAI RENÉ HARTMUT LAWONN

ILLUSTRATIVE VISUALIZATION OF MEDICAL DATA SETS



delivered by
EUROGRAPHICS DIGITAL LIBRARY
www.eg.org diglib.eg.org

ILLUSTRATIVE VISUALIZATION OF MEDICAL DATA SETS

DISSERTATION

zur Erlangung des akademischen Grades

Doktoringenieur (Dr.-Ing.)

angenommen durch die Fakultät für Informatik
der Otto-von-Guericke-Universität Magdeburg

von DIPL.-MATH. KAI RENÉ HARTMUT LAWONN

geb. am 04.10.1985, in Berlin

Gutachter

Prof. Dr. Bernhard Preim
Prof. Dr. Stefan Bruckner
Prof. Dr. Lars Linsen

Magdeburg, den 15. September 2014

ABSTRACT

The aim of an illustrative visualization method is to provide a simplified representation of a complex scene or object. Concave and convex regions are emphasized and the surface complexity is reduced by omitting unnecessary information. This abstraction is often preferred over fully illuminated scenes in a multitude of applications.

This thesis presents an overview of the state-of-the-art for feature lines. For this, the mathematical background of the differential geometry will be presented. Furthermore, this background will be adapted to triangulated surface meshes. This thesis will also present a smoothing algorithm, which smooths initial curves on triangulated surfaces. Additionally, an evaluation is shown to assess the quality of feature lines on anatomical data sets. Based on the evaluation, two conclusions in the field for medical applications were derived. From this point, this thesis presents two solutions in the field of illustrative visualization for medical data sets. A novel line drawing technique will be presented to illustrate surfaces. According to different requirements, this technique will be extended for molecular surfaces. In the field of vessel visualization with embedded blood flow, an adaptive visualization method will be introduced. This technique will also be extended to animated blood flow. Finally, this thesis shows different illustrative visualization concepts, which can be applied in various fields for depicting surface information.

ZUSAMMENFASSUNG

Das Ziel der illustrativen Visualisierung ist die vereinfachte Darstellung einer komplexen Szene oder eines komplexen Objektes. Hierbei wird versucht die Komplexität der Oberflächenstruktur zu vereinfachen indem unwichtige Informationen ausgelassen werden. Diese abstrakte Darstellung wird oft bevorzugt im Vergleich zu physikalisch korrekten voll ausgeleuchteten Szenen.

Diese Dissertation präsentiert den aktuellen Stand der Merkmalslinien. Hierfür wird der mathematische Hintergrund der Differentialgeometrie vorgestellt. Dies wird dann auf triangulierte Oberflächen angepasst. Weiterhin wird ein Algorithmus präsentiert, der ausgehend von einer initialen Kurve auf einer Oberfläche diese Kurve glättet. Anschließend wurde eine Evaluierung durchgeführt um die Qualität der Merkmalslinien auf anatomischen Oberflächen festzustellen. Diese Evaluierung leitete zwei Erkenntnisse für die medizinische Anwendung ab. Für jeden Teil wird eine Lösung für die medizinische Visualisierung präsentiert. Eine neue Strichzeichnung für die Illustration von Oberflächen wird vorgestellt. Diese Technik wird dann aufgrund verschiedener Anforderungen für Molekülоberflächen erweitert. Im Bereich der Gefäßvisualisierung mit eingeschlossenem Blutfluss wird eine adaptive Visualisierungstechnik gezeigt. Dieser Algorithmus wird dann im Folgenden für animierte Blutflussdaten erweitert. Schließlich zeigt die vorliegende Dissertation verschiedene Visualisierungskonzepte, die in verschiedenen Anwendungsbereichen benutzt werden können um Oberflächeninformationen zu vermitteln.

Und jedem Anfang wohnt ein Zauber inne,
Der uns beschützt und der uns hilft, zu leben.

— Hermann Hesse, Stufen

DANKSAGUNGEN

Die Danksagung ist meist ein schwieriges Unterfangen. Vielen Menschen werden gedankt, meist für die gute Betreuung, für freundschaftlichen Rat oder familiären Beistand. Der erste Dank gilt meistens dem Doktorvater wahrscheinlich aus Tradition und oder aus der geleisteten Unterstützung heraus. Diese Danksagung bricht diese Regel nicht unterstützt den Dank aber aus anderen Beweggründen. Natürlich möchte ich mich bei meinem Doktorvater Bernhard Preim zuerst bedanken für die tolle Unterstützung, für den vielen Zuspruch und für die Fähigkeit Menschen zu motivieren. Gerade in bewölkten wissenschaftlichen Tagen kann man in Bernhards Büro gehen und geht mit glücklicher Miene und neuer Motivation heraus. Meist geschieht dies ohne zu wissen, wie er es diesmal wieder geschafft hat. In allererster Linie möchte ich mich aber bei Bernhard für sein Vertrauen bedanken. Er hat mir die Möglichkeit gegeben bei ihm zu promovieren, obwohl er mich kaum kannte. Aus einer Initiativbewerbung heraus suchte er nach Wegen um mich zu beschäftigen. Und er hat es wieder einmal geschafft. Aus diesem Grund, gilt mein allergrößter Dank wirklich dir Bernhard und ich hoffe du hast deine Entscheidung niemals bereut. Danke!

Als nächstes möchte ich mich bei vielen Kollegen bedanken, zunächst einmal dafür, dass ich meine Kollegen auch Freunde nennen darf. Es macht so vieles leichter. Für die Unterstützung der GPU-Programmierung bedanke ich mich bei Tobias „GPU-Joe“ Mönch. Mein erstes illustratives Visualisierungspaper – ohne dich Tobias undenkbar. Und jedes Mal muss ich an unsere Debug-Ideen denken. „So und nun mappen wir die Krümmung mal auf den Rot-Kanal.“

Der nächste Dank gilt Rocco „Schocco“ Gasteiger. Rocco vielen Dank, dass du immer Zeit gefunden hast den einen oder anderen Bug zu finden. Und vielen Dank für deine gute Laune, die sehr zum positiven Büroklima beigetragen hat.

Einen weiteren Dank richte ich auch an Christoph „The Nvidia-Guy“ Kubisch. Christoph ich danke dir für deine Geduld und deine super schnellen Antworten.

Als nächstes danke ich auch Mathias „Neugerysm“ Neugebauer. Mathias unsere vielen Diskussionen haben mir wirklich Spaß gemacht. Ich danke dir auch, dass ich an vielen spannenden Projekten

mitarbeiten konnte. Und besonders danke ich dir für lustige Ablenkungen. Ach Mathias noch etwas: „Ja, ich habe mal darüber nachgedacht...“ :).

Ein weiterer Dank gilt Sylvia „Sylvus“ Glaßer für die spaßigen Projekte und für die Zusammenarbeit an einigen Publikationen.

Bei Paul „Klemmster“ Klemm möchte ich mich für die gute Laune und die Bereitschaft herumzualbern bedanken. Ich finde es gut, dass du dir einige Späße nicht verklemmen kannst.

For the sustained proof-reading, I am grateful to Stefanie Quade for your help.

Steffen „Dr. O“ Oeltze-Jafra, Junge, weißte Bescheid oder wie oder was?! :)

Alexandra „Evaluationsbär“ Baer ohne dich wäre der Büroalltag nur halb so lustig!

Bei meinem Banknachbarn Patrick „Mr. Ross“ Saalfeld bedanke ich mich für die kleinen Bürostreiche, die sehr zum Spaß an der Arbeit beitragen.

Benjamin „Der Ben“ Köhler, ich hoffe du hast bald einen neuen Trainingsplan für mich! :)

Ach ja und eh ich es vergesse Tobias „GPU-Günther“ Günther. Montag ist doch hoffentlich wieder der Sub-Tag (aber erst nach dem Sport).

Zu guter Letzt möchte ich mich natürlich noch bei meiner Familie (Susanne Thess-Lawonn, Michael Thess, Melissa Lawonn, Hartmut Knappe, Christa Knappe) für die Unterstützung und das Vertrauen bedanken. Besonders erwähnen möchte ich hier noch meine Mutter (Susanne Thess-Lawonn), die mir ständig bei den Schulaufgaben geholfen hat :). An anderer Stelle noch meinem Opa (Hartmut Knappe), mit dem ich am Telefon immer herumalbern kann und der nach wie vor immer ein offenes Ohr für mich hat.

Kai Lawonn, 15. September 2014

CONTENTS

Part I

PRELIMINARIES

1 Introduction

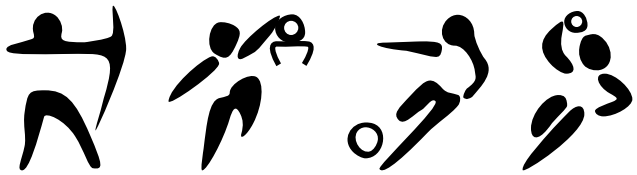
This section is partly based on:

Kai Lawonn and Bernhard Preim
Feature Lines for Illustrating Medical Surface Models:
Mathematical Background and Survey
Visualization in Medicine and Life Sciences
(in print), 2014

1 INTRODUCTION

1.1 INTRODUCTION

The application of illustrative visualization has increased in recent years. The principle goal behind the concept of illustrative visualization is a meaningful, expressive, and simplified depiction of a problem, a scene, or a situation. As an example, running people are represented by running stickmans, which can be seen in the Olympic games, and other objects become simplified line drawings, see Figure 1. More complex examples can be found in medical atlases. Most anatomical structures are painted and illustrated with pencils and pens. Gray's anatomy is one of the famous textbooks for medical teaching. Most of the other textbooks in this area orient to depict anatomy with art drawing, too.



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

Figure 1: Visual abstraction of the four Olympic disciplines: archery, basketball, football and handball in the style of pictograms of the Olympic Games 2012 in London.

Other than simplified representation, illustrative visualization is not restricted to these application fields. Illustrative techniques are essential for *focus-and-context* visualizations. Consider a scene with anatomical structures and one specific (important) structure. The specific structure may be strongly related to the surrounding objects. Therefore, hiding the other objects is not a viable option. In contrast, depicting all structures leads to visual clutter and optical distraction of the most important structures. Focus-and-context visualization is characterized by a few local regions that are displayed in detail and with emphasis techniques, such as a saturated color. Surrounding contextual objects are displayed in a less prominent manner to avoid distraction from focal regions. Medical examples are vessels with interior blood flow, livers with inner structures including vascular trees and possible tumors, proteins with surface representation and interior ribbon visualization. Focus-and-context visualization is not restricted to medical data. An example is the vehicle body and the interior devices.

The user or engineer needs the opportunity to illustrate all devices in the same context.

There are numerous illustration techniques. This dissertation is focused on a specific illustrative visualization category: line drawing. Line drawing techniques can be divided into two groups: *feature lines* and *hatching*. This thesis presents the application of line drawing techniques on medical data sets. First, feature lines will be analyzed and used to emphasize surfaces. Here, requirements and recommendations will be derived and the advantages and disadvantages will be discussed. Second, hatching methods were used to overcome the disadvantages, but exhibit other problems. Therefore, this thesis will guide the reader through established line drawing methods. It will list requirements and will show that these line drawing techniques are not able to fulfill these requirements. In an evaluation, two conclusions will be made that lead to different ways to resolve the disadvantages. Both directions are covered in this thesis. The first direction covers the disadvantages of feature lines and hatching. A new line drawing technique will be presented and its advantages will be confirmed in an evaluation. This technique will be extended and presented. The second direction is an adaptive shading, which uses the advantages of an established feature line method. The advantages were also confirmed in comparison to established shading techniques. This method will also be discussed and extended.

1.2 ORGANIZATION

The thesis provides the fundamentals of illustrative visualization in Chapter 2. This chapter provides publications according to line drawing techniques as well as publications that are related to illustrative visualization of medical surfaces. Furthermore, it covers the common publications, which are most related to this thesis and to the own contributions. Also, a mathematical prerequisite is presented. This chapter introduces the necessary fundamentals of differential geometry. Afterwards, these fundamentals will be adapted to triangulated surface meshes.

In Chapter 3, the data acquisition pipeline for vessel surfaces is presented. This will be important for the later contribution, which is related to vessel surfaces with interior blood flow. Furthermore, it presents a smoothing algorithm, which smoothes initial curves on triangulated surfaces.

In Chapter 4, several feature line methods are presented and compared. Furthermore, this chapter presents examples where feature lines can be used in the medical application. Additionally, a qualitative evaluation is provided to assess the suitability of feature lines on specific medical surfaces.

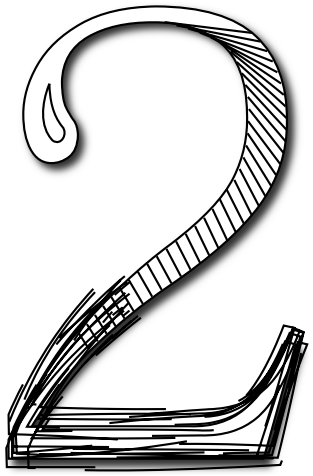
In Chapter 5, a line drawing technique is presented that overcomes the disadvantages of *feature lines* and *hatching* and combines their advantages. Furthermore, this chapter presents an evaluation to assess the quality of this method compared to established *hatching* methods on endoscopic views.

As the aforementioned technique has some limitations, a further improved line drawing technique is presented in Chapter 6. The technique overcomes the previous limitations. This method will be explained along with examples from molecular dynamic simulations. The specific requirements due to the dynamic character and the information needs of biologists are discussed.

In Chapter 7, a shading with a certain feature line method is presented. This is combined with an animation of the interior blood flow represented by pathlines.

Afterwards, a visualization is presented, which illustrates an aneurysm with its wall thickness in Chapter 8. An additional surface clustering is applied to gain an insight into the homogeneity of the wall thickness and wall-shear stress distribution on the surface.

Fundamentals of Computer-Generated Line Drawings



This section is partly based on:

Kai Lawonn and Bernhard Preim
Feature Lines for Illustrating Medical Surface Models:
Mathematical Background and Survey
Visualization in Medicine and Life Sciences
(in print), 2014

2

FUNDAMENTALS OF COMPUTER-GENERATED LINE DRAWINGS

This chapter provides the fundamentals of computer-generated line drawings as well as the mathematical fundamentals. In general, line drawings belong to the group of non-photorealistic rendering (NPR) techniques. NPR techniques are a widespread area for expressive visualizations. In contrast to photorealistic rendering, the focus is not to generate physically correct images. The expressive result is mostly inspired by artistic styles. An overview of the variety of NPR techniques can be found in [68, 75, 185]. Examples about the perceptual evaluation of different rendering techniques will be presented. This section will first give an overview of the most common line drawing techniques. According to Rautek et al. [165] these illustrative visualization techniques are called *low-level visual abstractions*. Here, the expressive techniques may change the layout of the surface or alter features to convey the communicative intent of the author.

High-level visual abstractions are techniques which convey relevance information for the examination. Rautek et al. mention different examples for high-level abstractions: interactive cutaways [26, 116], close-ups [24, 190], exploded views [25], or peel-aways [43] for volume data.

Dooley and Cohen presented one of the first line drawing techniques [54]. They proposed a system for the automatic generation of line drawings on surfaces. Also, different line styles were introduced to encode spatial relations. For example, wider lines are used for near regions and dashed lines denote hidden parts. Saito and Takahashi [171] introduced a rendering technique that generates contour lines. They used depth information as well as the contours to enhance the shaded result. Furthermore, curved hatching schemes are used to convey a spatial impression. Therefore, line drawing techniques can be divided in two groups: *feature lines* and *hatching*. Here, a short overview of feature lines as well as hatching methods will be presented. Afterwards, an overview of illustrative visualization in medical applications is provided. Different medical applications are listed where various illustrative visualization techniques are well used. Furthermore, this chapter provides an introduction to the differential geometry. Finally, this mathematical concept will be adapted to the discrete differential geometry, the area of discrete surface meshes.

2.1 VISUAL PERCEPTION

Perceptual studies are an important part in the assessment of illustrative visualization techniques. Such studies are useful for subsequent refinements. In the following, three different studies for the assessment of illustrative visualization techniques are presented.

Isenberg et al. [94] proposed an observational evaluation for the understanding and the assessment of NPR techniques compared to hand drawings. This study was conducted with three groups. The participants had to fulfill several tasks. In the first task, the participants were shown 30 illustrations ordered differently. Afterwards, they were asked to sort them in an arbitrary number of piles according to their own criteria. Then, the participants were asked a few questions regarding their personal opinion about the images. They should tell which image they like best / least and why. Further questions were raised regarding the purpose and the application of the images. The last task was to complete a questionnaire. As a result, Isenberg et al. pointed out several recommendations for NPR techniques. For example, they stated that the NPR techniques should avoid patterns and regularities if the goal was to generate hand drawn looking illustrations.

Cole et al. [41] conducted an extensive study about line drawings. The goal of the study was to assess where artists draw lines to convey 3D shapes. In this study, 29 artists' drawings of 12 different surfaces were analyzed. The evaluation was conducted in two tasks. The first task was about drawing the surfaces on a blank sheet. The second task was to recreate the same line drawings by tracing over a faint copy of the surface. Afterwards, the drawing result was scanned. Therefore, the artists could register the lines to the corresponding pixels to analyze the locations of the drawing. Finally, the images were used to analyze the line drawing positions. Several conclusions were stated.

- About 75% of the lines drawn by the artists are within 1 mm of the other line drawings.
- An indicator for line drawings is a large gradient in the image intensity.
- More interestingly, the authors claimed that in some cases the artists used line drawing criteria which are beyond local properties. Some lines were drawn on weak ridge and valley features, whereas other lines were omitted on stronger ridge and valley features.

Baer et al. [6] performed a quantitative evaluation for assessing the spatiality of different vessel visualization techniques. They used gauges on surfaces to measure the spatial impression. The participants were asked to adjust the gauge according to an imaginary normal. For this, the user had just a limited view of the 3D surface. This

ensures that the participants could not adjust the camera to view sideways. Otherwise, the normal could be adjusted easily according to the contour. Afterwards, Baer and colleagues could measure the deviation of the gauge compared to the computed normal. Therefore, a small deviation is an indicator for an illustrative visualization technique that can convey a reasonable spatial impression.

2.2 LINE DRAWING

Line drawing techniques depict surfaces with several lines. These techniques may be used to emphasize certain regions or to illustrate unessential regions and to understate them.

2.2.1 *Feature Lines*

Feature lines attempt to illustrate salient regions with single lines. For feature lines, image and object space methods have to be considered, which use an image or a surface model, respectively.

Image space methods use the image as an input. All calculations are performed on the rendered image where every pixel contains an RGB- or gray value. The image is usually convoluted with different kernels, e.g., Laplacian kernel. The Laplacian kernel is a discretized version of the continuous Laplace operator. It has the following structure:

$$\begin{pmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{pmatrix}.$$

Every pixel in an image is convoluted with this matrix, which means that all pixels obtain a new RGB value that is obtained by multiplying the pixel's RGB value by four and subtracting the four neighbored pixels RGB values. The final image highlights edges. The method works best, if the contrast of the edge compared to the surroundings is high. Kernel-based techniques have the advantage that they can be performed very fast. Through multi-processing or GPU implementation, several pixels can be processed simultaneously. Comprehensive overviews of image-based feature lines are given by Muthukrishnan and Radha [148], Nadernejad et al. [149], and Senthilkumaran and Rajesh [179]. The resulting feature lines are not frame-coherent and are represented by pixels in the image space. Thus, there is only limited control over the resulting line attributes, e.g., rendering style. The frame-coherence is an important requirement for the exploration of data and therefore, this thesis will not be focused on image-based approaches.

Object-based methods use the connectivity and the 3D vertex position of the surface model as input. Additional information (camera, light position, curvature) may be used to detect features. The extracted lines are located on the surface and are represented as explicit 3D lines. Thus, they are usually frame-coherent and any stylistic rendering technique can be applied. For an extensive list of line drawings and their applications, see Rusinkiewicz et al. [170] and Preim and Botha [164]. The feature line methods will be explained in more detail in Chapter 4.

The most important object-space lines are contours, which depict the strongest shape cues of the model. Unfortunately, contours cannot express all features being relevant for shape perception. Interrante et al. [91] proposed *ridges and valleys* as additional lines to convey the features.

Ridges and valleys are defined as the loci of points at which the principle curvature reaches an extremum in the principle direction. DeCarlo et al. [46] introduced *suggestive contours*, which convey shape by extending the contours of the surface mesh. These lines are defined as the set of minimum of the dot product between the surface normal and the tangential view vector. Unfortunately, objects without concave regions, e.g., most human organs, have no suggestive contours.

To resolve this problem, Judd et al. [99] presented *apparent ridges*, which extend the definition of ridges by using view-dependent curvature and a view-dependent principle curvature direction. Thus, these lines occur where the derivative of the maximum view-dependent curvature in direction of the associated principle view-dependent curvature direction equals zero. Besides the depiction of lines, Ni et al. [151] proposed view-dependent feature lines. They reduce the details in distant regions where the perception of fine details would be difficult.

Kolomenkin et al. [111] provided *demarcating curves* as the zero crossing of the normal curvature in the curvature gradient direction. They used higher-order derivatives and determined a $2 \times 2 \times 2$ rank-3 curvature tensor.

Xie et al. [209] introduced *photic extremum lines* (*PELs*). These feature lines are view-dependent and light-dependent as well. *PELs* are defined as the set of points where the variation of illumination in the gradient direction reaches a local maximum. *PELs* generation has been optimized by Zhang et al. [212] to reach real-time performance.

Furthermore, Zhang et al. [213] introduced *Laplacian lines*. They generalize the Laplacian of Gaussian edge detector to 3D surfaces. The *Laplacian lines* are defined as a set of points where the Laplacian of the illumination passes through zero. Furthermore, the gradient magnitude must be greater than a user-specified threshold.

2.2.2 *Hatching*

Hatching is another category of common illustrative visualization techniques. It emphasizes the surface by drawing lines mostly along principal curvature directions. The lines, drawn on different regions, are oriented depending on the shading of the model. Salisbury et al. [173] introduced an interactive framework for creating pen-and-ink illustrations. In this framework, the user paints with a collection of strokes in different patterns. Strothotte et al. [186] developed line drawing techniques for rendering images. A user can adjust the result by using primitives with various degree of precision. Therefore, the result may look like sketches, but still hand drawn. Winkenbach and Salesin [207] discussed the technical problems and solutions for computer-generated pen-and-ink illustrations. They extended the work for rendering free-form surfaces (B-spline surfaces, Nurbs, and surfaces of revolution) [208]. They used a controlled-density hatching to generate simultaneously toned lines. Furthermore, this technique is also used to extend the range of effects, e.g., highlights. Unfortunately, this method can only be applied to free-form surfaces. Salisbury et al. [174] introduced an interactive system for line drawings. They used a grayscale image and the user had to adjust the directional field. Afterwards, the strokes were automatically generated based on a user-specified set of example strokes. Girshick and Interrante [72] estimated principal directions on the surface mesh. Afterwards, lines were traced on the surface along the principal directions. Hertzmann and Zorin [83] described a method to illustrate smooth surfaces. First, they extracted the silhouette curves and separated two images of the area. These areas and the shading were used to define the hatching styles. Praun et al. [163] introduced a real-time hatching method which builds a lapped texture parametrization over the mesh that is aligned to a curvature-based direction field. Gasteiger et al. [65] presented a texture-based method to hatch anatomical structures. Buchin and Walther [29] presented a hatching scheme which can interactively change the stroke style. Zander et al. [211] used streamlines instead of textures to generate hatched lines based on curvature information. Gerl and Isenberg [69] presented a way to intuitively interact with the hatching illustration based on hand drawn examples. They used a scanned-in hatching picture as input and machine learning methods to learn a model of the drawing style.

2.2.3 *Conclusion*

In summary, the available feature line methods are not sufficient to depict arbitrary shapes, especially in the domain of anatomical structures. They highlight features, but local shape properties, which relate to curvature changes, are often not satisfyingly represented. The

latter is resolved by hatching methods, which have the drawbacks of being dependent on texturing or drawing too many hatching patterns. Thus, real-time performance cannot always be guaranteed in combination with high visual quality.

2.3 ILLUSTRATIVE VISUALIZATION FOR MEDICAL APPLICATION

In this section, we list different application fields where illustrative visualization is useful for effectively depicting medical image data. Especially line drawings were extensively used for medical visualization tasks, such as displaying tissue boundaries in volume data [30, 197], vascular structures [168], neck anatomy [114] and brain data [96]. Ebert and Rheingans [57] introduced a non-photorealistic approach to render volume data sets. Dong et al. [53] addressed a volume hatching, which takes the influence of the intensity values for the stroke generation into account. In the field of focus-and-context visualization and their application, an overview is given in [27]. Sikachev et al. [182] used a dynamic approach for the exploration of volumes based on focus-and-context rendering. Viola [202] used illustrative visualizations for volume rendering. Bruckner [23] described methods to generate illustrations from volumetric data. Fischer et al. [63] proposed illustrative visualizing tools to depict structures of hidden surfaces. The rendering style is tailored for understanding spatial relationships and for visualizing hidden objects. Born et al. [19] used illustrative techniques to depict stream surfaces. Their techniques are very useful for visualizing complex flow structures. In the area of brain data, Jainek et al. [96] suggested to use a hybrid visualization method to illustrate mesh and volume rendering. Their approach is efficient for the exploration in clinical research. Chu et al. [39] proposed a guideline of various rendering techniques. They combined, e.g., isophote-based line hatching and silhouette drawing for illustrative vascular visualization. Different rendering techniques for medical applications were presented by Tietjen et al. [195]. An example for illustrative visualization for liver surgery can be found in [80]. Haidacher et al. [78] introduced a statistical transfer-function, i.e., functions that map the voxels properties to optical properties, space which is used to define statistical transfer-function. This statistical transfer-function allows to visualize volume even in the presence of noise. Csebfalvi et al. [45] employed a novel volume rendering that ensures a fair contribution of different layers to different regions of the image space. They used different illumination directions for different materials, and view-dependently adjust the opacities to illustrate the volume.

Everts et al. [59] employed depth-dependent halos to illustrate dense lines. Tight line bundles are emphasized, whereas less structured lines are understated. Everts et al. [58] proposed illustrative line styles

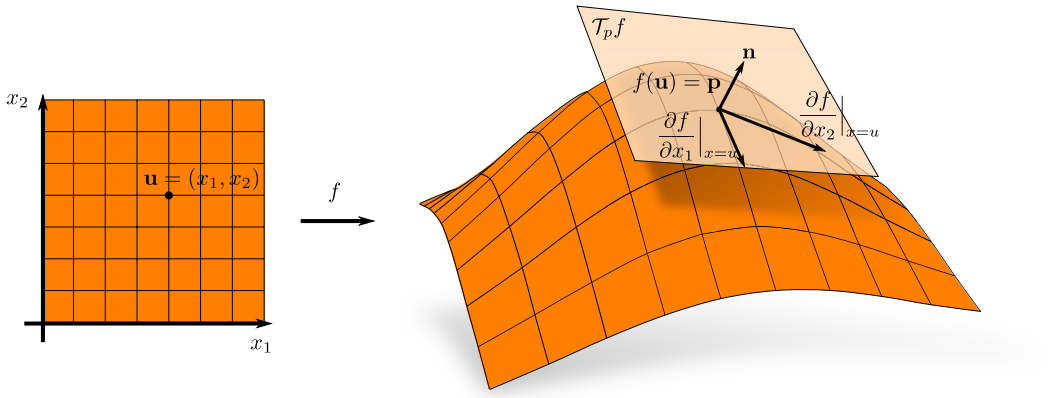


Figure 2: The basic elements of differential geometry. A parametric surface is given and the partial derivatives create the tangent space.

for visualizing streamlines. Svetachov et al. [189] used an illustrative visualization method to visualize DTI fiber tracts and their context. They oriented to pen-and-ink illustrations styles and adapt them.

A comprehensive overview of illustrative visualization in the field of flow visualization is given in [22]. Bruckner et al. [28] presented a framework where rendered 3D layers can be composed. Their method can be used to illustrate different layers of anatomic structures, e.g., the heart with vessels and muscles. Gerl et al. [70] employed an interactive approach for the explicit specification of semantics for volume visualization.

2.4 GEOMETRY BACKGROUND

This section presents the fundamentals of differential geometry for feature line generation, which will be crucial for the further sections. The basic terms and properties will be presented. This section is inspired by differential geometry books [50, 51, 117].

2.4.1 *Basic Prerequisites*

A surface $f\colon I \subset \mathbb{R}^2 \to \mathbb{R}^3$ is called a parametric surface if f is an *immersion*. An *immersion* means that all partial derivatives $\frac{\partial f}{\partial x_i}$ are injective at each point. The further calculations are mostly based on the tangent space of a surface. The tangent space $\mathcal{T}_p f$ of f is defined as the linear combination of the partial derivatives of f:

$$\mathcal{T}_p f := \operatorname{span}\left\{ \frac{\partial f}{\partial x_1}\Big|_{\mathbf{x}=\mathbf{u}}, \frac{\partial f}{\partial x_2}\Big|_{\mathbf{x}=\mathbf{u}} \right\}.$$

Here, span is the space of all linear combinations. Formally it is defined as: $\operatorname{span}\{\mathbf{v}_1, \mathbf{v}_2\} := \{\alpha\mathbf{v}_1 + \beta\mathbf{v}_2 \mid \alpha, \beta \in \mathbb{R}\}$. With the tangent space, a normalized normal vector $\mathbf{n}$ can be defined. The (normalized) normal vector $\mathbf{n}(\mathbf{u})$ at $\mathbf{p} = f(\mathbf{u})$ is defined such that for all elements

$\mathbf{v} \in \mathcal{T}_p f$ the equation $\langle \mathbf{v}, \mathbf{n}(\mathbf{u}) \rangle = 0$ holds, where $\langle ., . \rangle$ denotes the canonical Euclidean dot product. Therefore, $\mathbf{n}(\mathbf{u})$ is defined as:

$$\mathbf{n}(\mathbf{u}) = \frac{\frac{\partial f}{\partial x_1}\Big|_{\mathbf{x}=\mathbf{u}} \times \frac{\partial f}{\partial x_2}\Big|_{\mathbf{x}=\mathbf{u}}}{\left\| \frac{\partial f}{\partial x_1}\Big|_{\mathbf{x}=\mathbf{u}} \times \frac{\partial f}{\partial x_2}\Big|_{\mathbf{x}=\mathbf{u}} \right\|}.$$

This is also called the Gauss map. Figure 2 depicts the domain of a parametric surface as well as the tangent space $\mathcal{T}_p f$ and the normal $\mathbf{n}$.

2.4.2 *Curvature*

The curvature is a fundamental property to identify salient regions of a surface that should often be conveyed by feature lines. Colloquially spoken, it is a measure of how far the surface bends at a certain point. If a person stands on a sphere at a specific point, it does not matter in which direction he goes, the bending will be the same. If the person stands on a plane at a specific point, he can go in any direction, there will be no bending. Without knowing any measure of the curvature, one can state that a plane has zero curvature and that a sphere with a small radius has a higher curvature than a sphere with a higher radius. This is due to the fact that a sphere with an increasing radius becomes locally more a plane. Intuitively, the curvature depends also on the walking direction. On a cylinder, there is a bending in one direction but not in the other. Painting the trace of a walk on the surface and view it in 3D space, we could treat this as a 3D curve. The definition of the curvature of a curve may be adapted to the curvature of a surface. The adaption of these concepts is explained in the following. Let $c\colon I \subset \mathbb{R} \to \mathbb{R}^3$ be a 3D parametric curve with $\|\frac{dc}{dt}\| = 1$. The property of constant length of the derivative is called arc length or natural parametrization. One can show that such a parametrization exists for each continuous, differentiable curve that is an immersion. So, for the measuring of the size of bending, the norm of the second derivative of the curve can be used. Therefore, the (absolute) curvature $\kappa(t)$ at a time step t is defined as:

$$\kappa(t) = \left\| c''(t) \right\|.$$

To determine the curvature on a certain point of the surface in a specific direction, a curve can be employed and its curvature can be calculated. This approach is imperfect because curves that lie in a plane can have non-vanishing curvature, e.g., a circle, whereas it was claimed to have zero curvature on a planar surface. Therefore, the second derivatives of the curve have to be distinguished regarding which part contributes to the tangent space and which contributes to

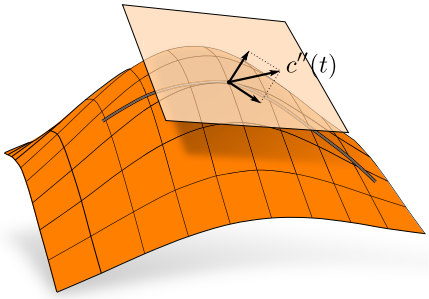


Figure 3: The curve's second derivative is decomposed into the tangential and normal part.

the normal part of the surface. Decomposing the second derivative of the curve into tangential and normal part of the surface yields:

$$c''(t) = \underbrace{\text{proj}_{\mathcal{T}_p f}\, c''(t)}_{\text{tangential part}} + \underbrace{\langle c''(t), \mathbf{n}\rangle \mathbf{n}}_{\text{normal part}},$$

where $c(t) = \mathbf{p}$ and $\text{proj}_E\, \mathbf{x}$ means the projection of the point $\mathbf{x}$ onto the space E, see Figure 3. The curvature $\kappa_c(\mathbf{p})$ of the surface at $\mathbf{p}$ along the curve c is defined as the coefficient of the normal part:

$$\kappa_c(p) = \langle c''(t), \mathbf{n}\rangle. \tag{1}$$

Hence, with $c'(t) \in \mathcal{T}_p f$ it follows: $\langle c'(t), \mathbf{n}\rangle = 0$. Deriving the last equation yields:

$$\frac{d}{dt}\langle c'(t), \mathbf{n}\rangle = 0$$
$$\frac{d}{dt}\langle c'(t), \mathbf{n}\rangle = \langle c''(t), \mathbf{n}\rangle + \langle c'(t), \frac{\partial \mathbf{n}}{\partial t}\rangle.$$

This yields:

$$\langle c''(t), \mathbf{n}\rangle = -\langle c'(t), \frac{\partial \mathbf{n}}{\partial t}\rangle.$$

Combining this equation with Equation 1 yields

$$\kappa_{c'(t)}(\mathbf{p}) = -\langle c'(t), \frac{\partial \mathbf{n}}{\partial t}\rangle. \tag{2}$$

Thus, the curvature of a surface at a specific point in a certain direction can be calculated with a theorem by Meusnier. The vectors $\mathbf{v}, \mathbf{w}$ at $\mathbf{p}$ are called the maximal / minimal principle curvature directions of the maximal and minimal curvature, if $\kappa_{\mathbf{v}}(\mathbf{p}) \geqslant \kappa_{\mathbf{v}'}(\mathbf{p})$, $\kappa_{\mathbf{w}}(\mathbf{p}) \leqslant \kappa_{\mathbf{v}'}(\mathbf{p})$ for all directions $\mathbf{v}' \in \mathcal{T}_p f$. If such a minimum and maximum exists, then $\mathbf{v}$ and $\mathbf{w}$ are perpendicular, see Section 2.4.5 for a proof. To determine the curvature in direction $\mathbf{u}$, the principle curvature directions $\mathbf{u}, \mathbf{v}, \mathbf{w}$ must be normalized. Then, $\kappa_{\mathbf{u}}(\mathbf{p})$ can be determined by:

$$\kappa_{\mathbf{u}}(\mathbf{p}) = \langle \mathbf{u}, \mathbf{v}\rangle^2 \kappa_{\mathbf{v}}(\mathbf{p}) + \langle \mathbf{u}, \mathbf{w}\rangle^2 \kappa_{\mathbf{w}}(\mathbf{p}). \tag{3}$$

The coefficients of the curvature are extracted by the decomposition of the principle curvature directions with the vector $\mathbf{u}$.

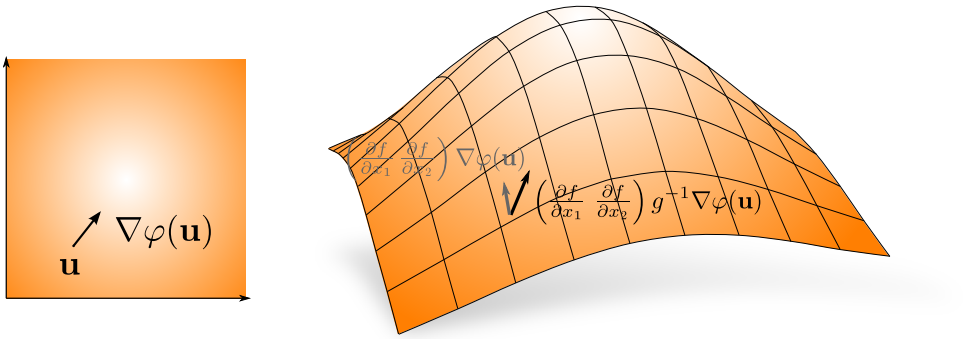


Figure 4: Given: a scalar field in the domain. Determining the gradient and using it as coefficient for the basis tangent vectors leads to the wrong result (gray arrow in the right image). Balancing the distortion with the inverse of the metric tensor yields the correct gradient on the surface (black).

2.4.3 *Covariant Derivative*

The essence of the feature line generation is the analysis of local variations in a specific direction, i.e., the covariant derivative. Therefore, the covariant derivative is a crucial concept for feature line methods. Consider a scalar field on a parametric surface $\varphi\colon f(I) \to \mathbb{R}$. One can imagine this scalar field as a heat or pressure (as well as a curvature) distribution. The directional derivative of φ in direction $\mathbf{v}$ can be written as $D_{\mathbf{v}}\varphi$ and is defined by:

$$D_{\mathbf{v}}\varphi(\mathbf{x}) = \lim_{h\to 0} \frac{\varphi(\mathbf{x}+h\mathbf{v}) - \varphi(\mathbf{x})}{h}.$$

If φ is differentiable at $\mathbf{x}$, the directional derivative can be simplified:

$$D_{\mathbf{v}}\varphi(\mathbf{x}) = \langle \nabla\varphi(\mathbf{x}), \mathbf{v}\rangle,$$

where ∇ denotes the gradient. The gradient is an operator applied to a scalar field and results in a vector field. To extend the definition of the derivative to an arbitrary surface, first the gradient of surfaces must be defined. In the following, the *covariant derivative* is used. The standard directional derivative results in a vector which lies somewhere in the 3D space, whereas the covariant derivative is restricted to stay in the tangent space of the surface. The gradient is a 2D vector. Actually, a 3D vector in the tangent space of the surface is needed. Here, the gradient is employed and used as coefficients of the tangential basis. Unfortunately, this leads to wrong results because of the distortions of the basis of the tangent space, see Figure 4. The basis is not necessarily an orthogonal normalized basis as in the domain space and can therefore lead to distortions of the gradient on the surface.

One way to calculate this vector is to use the plain scalar field $\varphi\colon \mathbb{R}^3 \to \mathbb{R}$. Afterwards, the gradient in 3D space is attained and

projected on the tangent space. However, the gradient of $\varphi\colon \mathbb{R}^2 \to \mathbb{R}$ in the domain of a parametric surface is used. It compensates the length distortion such that it can be used as coordinates with the basis in the tangent space. When multiplying the gradient with the i-th basis vector, one obtains the partial derivative of φ with x_i. Hence, it is known that the 3D gradient $\nabla\varphi$ lies in the tangent space. Therefore, it can be represented as a linear combination of $\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}$ with coefficients α, β:

$$\nabla\varphi = \alpha \cdot \frac{\partial f}{\partial x_1} + \beta \cdot \frac{\partial f}{\partial x_2}.$$

Multiplying both sides with the basis vectors and using the relation $\frac{\partial \varphi}{\partial x_i} = \langle \nabla\varphi, \frac{\partial f}{\partial x_i} \rangle$, an equation system is obtained:

$$\begin{aligned}
\begin{pmatrix} \frac{\partial \varphi}{\partial x_1} \\ \frac{\partial \varphi}{\partial x_2} \end{pmatrix} &= \begin{pmatrix} \alpha \cdot \langle \frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_1} \rangle + \beta \cdot \langle \frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2} \rangle \\ \alpha \cdot \langle \frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2} \rangle + \beta \cdot \langle \frac{\partial f}{\partial x_2}, \frac{\partial f}{\partial x_2} \rangle \end{pmatrix} \\
&= \underbrace{\begin{pmatrix} \langle \frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_1} \rangle & \langle \frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2} \rangle \\ \langle \frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2} \rangle & \langle \frac{\partial f}{\partial x_2}, \frac{\partial f}{\partial x_2} \rangle \end{pmatrix}}_{g :=} \begin{pmatrix} \alpha \\ \beta \end{pmatrix}.
\end{aligned}$$

The matrix g is called the *metric tensor*. This tensor describes the length and area distortion from $\mathbb{R}^2$ to the surface. The last equation yields the coefficients α, β when multiplied with the inverse of g:

$$\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = g^{-1} \begin{pmatrix} \frac{\partial \varphi}{\partial x_1} \\ \frac{\partial \varphi}{\partial x_2} \end{pmatrix}.$$

This leads to a general expression of the gradient for a scalar field $\varphi\colon \mathbb{R}^n \to \mathbb{R}$:

$$\nabla\varphi = \sum_{i,j=1}^{n} \left(g^{ij} \frac{\partial \varphi}{\partial x_j} \right) \frac{\partial}{\partial x_i}, \tag{4}$$

where g^{ij} is the i,j-th matrix entry from the inverse of g and $\frac{\partial}{\partial x_i}$ means the basis. Now, the covariant derivative of a scalar field can be calculated by first determining its gradient and afterwards using the dot product:

$$D_w\varphi = \langle \nabla\varphi, w \rangle. \tag{5}$$

2.4.4 *Laplace-Beltrami Operator*

The Laplace-Beltrami operator is needed for a specific feature line method and will therefore be introduced. The Laplace operator is defined as a composition of the *gradient* and the *divergence*. When interpreting the vector field as a flow field, the divergence is a measure of

how much more flow leaves a specific region than flow enters. In the Euclidean space, the divergence $\operatorname{div}\Phi$ of a vector field $\Phi\colon \mathbb{R}^n \to \mathbb{R}^n$ is the sum of the partial derivatives of the components Φ_i:

$$\operatorname{div}\Phi = \sum_{i=1}^{n} \frac{\partial}{\partial x_i}\Phi_i.$$

The computation of the divergence for a vector field $\Phi\colon \mathbb{R}^n \to \mathbb{R}^n$ in Euclidean space is straightforward. However, for the computation of the divergence of an arbitrary surface the length and area distortions have to be considered. Without giving a derivation of the divergence, the components Φ_i of the vector field have to be weighted by the square root of the determinant $\sqrt{|g|}$ of the metric tensor g before taking the derivative. The square root of the determinant of g describes the distortion change from the Euclidean space to the surface. Formally, the divergence of a vector field $\Phi\colon \mathbb{R}^n \to \mathbb{R}^n$ with a given metric tensor g is given by:

$$\operatorname{div}\Phi = \frac{1}{\sqrt{|g|}} \sum_{i=1}^{n} \frac{\partial}{\partial x_i}\left(\sqrt{|g|}\ \Phi_i\right). \tag{6}$$

Given the definition of the gradient and the divergence, both operators can be composed to obtain the Laplace-Beltrami operator $\Delta\varphi$ of a scalar field $\varphi\colon \mathbb{R}^n \to \mathbb{R}$ on surfaces:

$$\Delta\varphi = \operatorname{div}\nabla\varphi = \frac{1}{\sqrt{|g|}} \sum_{i,j=1}^{n} \frac{\partial}{\partial x_i}\left(\sqrt{|g|}\, g^{ij}\ \frac{\partial\varphi}{\partial x_j}\right). \tag{7}$$

2.4.5 *Shape Operator*

The curvature of a parametric surface at a specific point $\mathbf{p}$ in a certain direction can be determined by Equation 2:

$$\kappa_{c'(t)}(\mathbf{p}) = -\langle c'(t), \frac{\partial \mathbf{n}}{\partial t}\rangle,$$

recall Section 2.4.2. Actually, this means that the curvature in the direction $c'(t)$ is a measure of how much the normal changes in this direction, too. Given is $\mathbf{v} \in \mathcal{T}_{\mathbf{p}}f$ with $\mathbf{p} = f(\mathbf{u})$ and $|\mathbf{v}| = 1$. Then, the coefficients α, β of $\mathbf{v}$ with the basis $\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}$ can be determined:

$$\begin{pmatrix}\alpha\\ \beta\end{pmatrix} = g^{-1}\begin{pmatrix}\langle \mathbf{v}, \frac{\partial f}{\partial x_1}\rangle\\ \langle \mathbf{v}, \frac{\partial f}{\partial x_2}\rangle\end{pmatrix}.$$

The coefficients (α, β) are used to determine the derivative of $\mathbf{n}$ along $\mathbf{v}$ by using the 2D curve $\tilde{c}(t) = \mathbf{u} + t\binom{\alpha}{\beta}$ yielding:

$$D_{\mathbf{v}}\mathbf{n} := \frac{d}{dt}\mathbf{n}(\tilde{c}(t)).$$

Defining $S(\mathbf{v}) := -D_{\mathbf{v}}\mathbf{n}$ yields a linear operator. This operator is called *Shape Operator* (also *Weingarten Map* or *Second Fundamental Tensor*). One can see that $S(\frac{\partial f}{\partial x_i}) = \frac{\partial \mathbf{n}}{\partial x_i}$ holds. Note that this operator can directly operate on the 3D space with a 3D vector in the tangent space, as well as with the 2D space with the coefficients of the basis. Therefore, it can be represented by a matrix S. Recall Equation 2, substituting c' with $\mathbf{v}$ and $\frac{\partial \mathbf{n}}{\partial t}$ by $S\mathbf{v}$ yields:

$$\kappa_{\mathbf{v}}(\mathbf{p}) = \langle \mathbf{v}, S\mathbf{v} \rangle.$$

It has to be shown that the principle curvature directions are the eigenvectors of S. Assuming $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{R}^2$ are the normalized eigenvectors with the eigenvalues $\lambda_1 \geqslant \lambda_2$. Every normalized vector $\mathbf{w}$ can be written as a linear combination of $\mathbf{v}_1, \mathbf{v}_2$: $\mathbf{w} = \alpha\mathbf{v}_1 + \beta\mathbf{v}_2$ with $\|w\| = \|\alpha\mathbf{v}_1 + \beta\mathbf{v}_2\| = \alpha^2 + \beta^2 + 2\alpha\beta\langle \mathbf{v}_1, \mathbf{v}_2 \rangle = 1$. This yields:

$$\kappa_{\mathbf{w}}(\mathbf{p}) = \langle \mathbf{w}, S\mathbf{w} \rangle = \frac{1}{2}[(\alpha^2 - \beta^2)(\lambda_1 - \lambda_2) + \lambda_1 + \lambda_2]. \tag{8}$$

One can see from Equation 8 that $\kappa_{\mathbf{w}}(\mathbf{p})$ reaches a maximum if $\beta = 0$, $\alpha = 1$, and a minimum if $\alpha = 0$, $\beta = 1$. If the eigenvalues (curvatures) are not equal, the principle curvature directions are perpendicular. For this, it needs to be shown that S is a self-adjoint operator. Thus, the equation $\langle S\mathbf{v}, \mathbf{w} \rangle = \langle \mathbf{v}, S\mathbf{w} \rangle$ holds. This is shown by using the property $\langle \mathbf{n}, \frac{\partial f}{\partial x_i} \rangle = 0$ and derive this with x_j:

$$\langle \frac{\partial \mathbf{n}}{\partial x_j}, \frac{\partial f}{\partial x_i} \rangle + \langle \mathbf{n}, \frac{\partial^2 f}{\partial x_i \partial x_j} \rangle = 0.$$

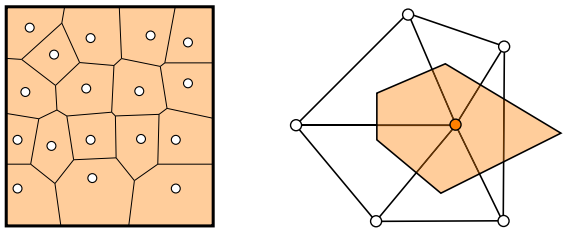
Next, it will be proven that S is a self-adjoint operator with the basis $\frac{\partial f}{\partial x_i}$:

$$\begin{aligned}
\langle S(\frac{\partial f}{\partial x_i}), \frac{\partial f}{\partial x_j} \rangle &= \langle -\frac{\partial \mathbf{n}}{\partial x_i}, \frac{\partial f}{\partial x_j} \rangle \\
&= \langle \mathbf{n}, \frac{\partial^2 f}{\partial x_i \partial x_j} \rangle \\
&= \langle -\frac{\partial \mathbf{n}}{\partial x_j}, \frac{\partial f}{\partial x_i} \rangle \\
&= \langle S(\frac{\partial f}{\partial x_j}), \frac{\partial f}{\partial x_i} \rangle.
\end{aligned}$$

Now, it will be shown that the eigenvectors (principle curvature directions) are perpendicular if the eigenvalues (curvatures) are different:

$$\lambda_1 \langle \mathbf{v}_1, \mathbf{v}_2 \rangle = \langle S\mathbf{v}_1, \mathbf{v}_2 \rangle = \langle \mathbf{v}_1, S\mathbf{v}_2 \rangle = \lambda_2 \langle \mathbf{v}_1, \mathbf{v}_2 \rangle.$$

The equation is only true if $\mathbf{v}_1, \mathbf{v}_2$ are perpendicular (and $\lambda_1 \neq \lambda_2$ holds).



(a) Points in 2D

(b) Points on a surface mesh

Figure 5: The Voronoi diagram of different settings. In (a) a Voronoi diagram of a set of points is determined. In (b) the Voronoi area is calculated. If one of the triangles is obtuse, the area leaves the triangle.

2.5 DISCRETE DIFFERENTIAL GEOMETRY

This section adapts the continuous differential geometry to discrete differential geometry, the area of polygonal meshes that approximate continuous geometries. The following notation is used in the remainder of this section. Let $M \subset \mathbb{R}^3$ be a triangulated surface mesh. The mesh consists of vertices $i \in V$ with associated positions $\mathbf{p}_i \in \mathbb{R}^3$, edges $E = \{(i,j) \mid i,j \in V\}$, and triangles $T = \{(i,j,k) \mid (i,j),(j,k),(k,i) \in E\}$. The vector $\mathbf{n}_i$ means the normalized normal vector at vertex i. Furthermore, $\mathcal{N}(i)$ denotes the neighbors of i. So, for every $j \in \mathcal{N}(i)$, $(i,j) \in E$ holds. Furthermore, the following notation is used regarding triangles $\triangle = (i,j,k)$ with vertices $\mathbf{p}_i, \mathbf{p}_j, \mathbf{p}_k$, and the edges are defined as $\mathbf{e}_1 = \mathbf{p}_i - \mathbf{p}_j, \mathbf{e}_2 = \mathbf{p}_j - \mathbf{p}_k, \mathbf{e}_3 = \mathbf{p}_k - \mathbf{p}_i$.

2.5.1 *Voronoi Area*

The term *Voronoi area* is important for the determination of the curvature. So, given are points in a 2D space. Every point is spread out in equal speed. If two fronts collide, they stop to spread out further at this region. After all fronts stopped, every point lies in a region that is surrounded by a front. This region is called a *Voronoi region*. Formally, given distinct points $\mathbf{x}_i \in \mathbb{R}^2$ in the plane, the Voronoi region for the point $\mathbf{x}_k$ is defined as the set of points $V(\mathbf{x}_k)$ with

$$V(\mathbf{x}_k) = \{\mathbf{x} \in \mathbb{R}^2 \,:\, \|\mathbf{x} - \mathbf{x}_k\| \leqslant \|\mathbf{x} - \mathbf{x}_j\|,\, j \neq k\}.$$

See Figure 5a for an example of a Voronoi diagram. To obtain the Voronoi area of a vertex on a surface mesh, the Voronoi area of each incident triangle is accumulated. The Voronoi area calculation is based on the method by Meyer et al. [140]. In case of a non-obtuse triangle, the Voronoi area at $\mathbf{p}_i$ is determined by the perpendicular bisector of the edges incident to $\mathbf{p}_i$. The point of intersection, the midpoint

of the incident edges, and the point itself define the endpoints of the Voronoi area. The triangle area of the Voronoi region equals:

$$\mathcal{A}_{\triangle}(\mathbf{p}_i) = \frac{1}{8}\Big(\|\mathbf{e}_1\|^2 \cdot \cot(\mathbf{e}_2, \mathbf{e}_3) + \|\mathbf{e}_3\|^2 \cdot \cot(\mathbf{e}_1, \mathbf{e}_2)\Big).$$

In case of an obtuse triangle, the Voronoi area is equal half of the triangle area if the angle at $\mathbf{p}_i$ is obtuse. Otherwise it is a quarter of the triangle area, see Figure 5b.

2.5.2 *Discrete Curvature*

Several approaches exist to approximate the curvatures. Some methods try to fit an analytic surface (higher order polynomials) to the mesh and determine the curvatures analytically [32, 74]. Another approach estimates the normal curvature along edges first and then estimates the shape operator [38, 79, 140, 156, 192]. Mostly, methods that fit an analytic surface are similar to the normal curvature estimation approaches. The first family of techniques fit parabolas, whereas the second techniques fit circles. This implies instability in case that neighbored vertices are close to a pair of intersecting lines. Other approaches are based on the calculation of the shape operator S [2, 40, 169]. The curvature estimation according to [169] is used in the following. Other approaches as [40] may produce large errors, e.g., on a sphere with radius 1, the curvature deviates strongly in areas of densely-spaced vertices. Increasing the resolution does not decrease the error. According to Rusinkiewicz [169], the curvature computation is based on the shape operator S. After S is determined on a triangle basis, it is adapted to vertices. The operator $S\mathbf{v}$ yields the change of the normal in the direction of $\mathbf{v}$:

$$S\mathbf{v} = D_{\mathbf{v}}\mathbf{n}.$$

This property is used to assess S for each triangle. When applying S to the edge $\mathbf{e}_1$, it should result in $\mathbf{n}_i - \mathbf{n}_j$ because of the change of the normals along the edge. A basis of the tangent space of the triangle is needed:

$$\tilde{\mathbf{e}}_1 = \frac{\mathbf{e}_1}{|\mathbf{e}_1|}, \qquad \tilde{\mathbf{e}}_2 = \frac{\mathbf{e}_2}{|\mathbf{e}_2|}.$$

Afterwards, the orthogonal normalized basis vectors $\mathbf{x}_{\triangle}, \mathbf{y}_{\triangle}$ are determined by:

$$\mathbf{x}_{\triangle} \coloneqq \tilde{\mathbf{e}}_1, \qquad \mathbf{y}_{\triangle} \coloneqq \frac{\mathbf{x}_{\triangle} \times (\tilde{\mathbf{e}}_2 \times \mathbf{x}_{\triangle})}{\|\mathbf{x}_{\triangle} \times (\tilde{\mathbf{e}}_2 \times \mathbf{x}_{\triangle})\|}. \tag{9}$$

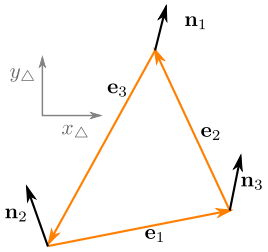


Figure 6: The shape operator estimation is based on a local coordinate system, the edges and the normals.

Applying the aforementioned property of the shape operator to all edges according to the basis leads to the following equation system:

$$\begin{aligned} S\begin{pmatrix}\langle \mathbf{e}_1, \mathbf{x}_\triangle\rangle \\ \langle \mathbf{e}_1, \mathbf{y}_\triangle\rangle\end{pmatrix} &= \begin{pmatrix}\langle \mathbf{n}_i - \mathbf{n}_j, \mathbf{x}_\triangle\rangle \\ \langle \mathbf{n}_i - \mathbf{n}_j, \mathbf{y}_\triangle\rangle\end{pmatrix} \\ S\begin{pmatrix}\langle \mathbf{e}_2, \mathbf{x}_\triangle\rangle \\ \langle \mathbf{e}_2, \mathbf{y}_\triangle\rangle\end{pmatrix} &= \begin{pmatrix}\langle \mathbf{n}_j - \mathbf{n}_k, \mathbf{x}_\triangle\rangle \\ \langle \mathbf{n}_j - \mathbf{n}_k, \mathbf{y}_\triangle\rangle\end{pmatrix} \\ S\begin{pmatrix}\langle \mathbf{e}_3, \mathbf{x}_\triangle\rangle \\ \langle \mathbf{e}_3, \mathbf{y}_\triangle\rangle\end{pmatrix} &= \begin{pmatrix}\langle \mathbf{n}_k - \mathbf{n}_i, \mathbf{x}_\triangle\rangle \\ \langle \mathbf{n}_k - \mathbf{n}_i, \mathbf{y}_\triangle\rangle\end{pmatrix}, \end{aligned} \tag{10}$$

see Figure 6 for an illustration. Here, three unknowns (the matrix entries of the symmetric matrix $S = \left(\begin{smallmatrix} e & f \\ f & g \end{smallmatrix}\right)$) and six linear equations are given. Thus, a least square method can be applied to fit the shape operator to approximate the curvature for each triangle. Next, S is calculated for each vertex of the mesh. As the triangle basis normally differs from each vertex tangent space basis, a transformation of the shape operator according to the new coordinate system is necessary. First, assuming that the normal $\mathbf{n}_\triangle$ of the face is equal to the incident vertex normal $\mathbf{n}_i$. Hence, the basis $(\mathbf{x}_\triangle, \mathbf{y}_\triangle)$ of the triangle is coplanar to the basis $(\mathbf{x}_i, \mathbf{y}_i)$ of the incident vertex i. Assuming the shape operator given in the vertex basis, then the entries can be determined by:

$$\begin{aligned} e_p &= \begin{pmatrix}1 & 0\end{pmatrix}\begin{pmatrix}e_p & f_p \\ f_p & g_p\end{pmatrix}\begin{pmatrix}1 \\ 0\end{pmatrix} = \mathbf{x}_i^\top S \mathbf{x}_i \\ f_p &= \begin{pmatrix}1 & 0\end{pmatrix}\begin{pmatrix}e_p & f_p \\ f_p & g_p\end{pmatrix}\begin{pmatrix}0 \\ 1\end{pmatrix} = \mathbf{x}_i^\top S \mathbf{y}_i \\ g_p &= \begin{pmatrix}0 & 1\end{pmatrix}\begin{pmatrix}e_p & f_p \\ f_p & g_p\end{pmatrix}\begin{pmatrix}0 \\ 1\end{pmatrix} = \mathbf{y}_i^\top S \mathbf{y}_i. \end{aligned}$$

As the shape operator is determined in the basis $(\mathbf{x}_\triangle, \mathbf{y}_\triangle)$, the basis of the vertex can be determined by expressing the new coordinate system with the old basis $\mathbf{x}_i = \alpha\mathbf{x}_\triangle + \beta\mathbf{y}_\triangle$:

$$\alpha = \langle \mathbf{x}_i, \mathbf{x}_\triangle \rangle$$
$$\beta = \langle \mathbf{x}_i, \mathbf{y}_\triangle \rangle.$$

The entry e_p can be determined by:

$$e_p = \begin{pmatrix} \alpha & \beta \end{pmatrix} S \begin{pmatrix} \alpha \\ \beta \end{pmatrix}. \tag{11}$$

The other entries can be calculated analogously. For the second case, the coordinate system of the triangle is rotated around the cross product of the normal such that the basis of the vertex and the triangle are coplanar. Finally, this can be used to determine the shape operator of the vertices. The shape operators for all incident triangles of a vertex are determined. Afterwards, the coordinate systems of the triangles are rotated to be coplanar with the basis of the vertex. Next, the shape operator is re-expressed in terms of the basis of the vertex. Then, the shape operator is weighted according to the Voronoi area of the triangle and accumulated. Finally, the accumulated shape operator is divided by the sum of the weights. The eigenvalues provide the principle curvatures, and the eigenvectors give the principle curvature directions according to the basis. The pseudo-code 1 summarizes the algorithm.

Algorithm 1 Pseudo-code for the curvature estimation.

```
for each triangle:
   Build basis accord. to Eq. 9
   Determine S accord. to Eq. 10
   for each vertex incident to the triangle:
      Rotate the triangle basis to the vertex basis
      Determine S in the new basis accord. to Eq. 11
      Add this tensor weighted by the Voronoi area
   end
end
for each vertex:
   Divide S by the sum of the weights
   Determine the eigenvalues and eigenvectors
end
```

Please note that this algorithm can be generalized to obtain higher-order derivatives. It can be used to determine the derivative of the

curvature as it is important for a specific feature line method. Formally, the derivative of the shape operator has the form:

$$C = \begin{pmatrix} D_v S & D_w S \end{pmatrix} = \left(\begin{pmatrix} a & b \\ b & c \end{pmatrix} \begin{pmatrix} b & c \\ c & d \end{pmatrix} \right). \tag{12}$$

For the determination of the change of the curvature in direction $\mathbf{u}$, the tensor C has to be multiplied multiple times:

$$D_{\mathbf{u}}\kappa = \langle \mathbf{u}, \begin{pmatrix} D_v S \cdot \mathbf{u} & D_w S \cdot \mathbf{u} \end{pmatrix} \cdot \mathbf{u} \rangle. \tag{13}$$

2.5.3 *Discrete Covariant Derivative*

The discrete covariant derivative is an important tool for assessing the local deviation of an underlying scalar field. The direction of the change as well as the strength of the change can be determined by the covariant derivative. First, a linear 2D scalar field $\varphi(\mathbf{x}) = \alpha \cdot x_1 + \beta \cdot x_2 + \gamma$ and its gradient is considered:

$$\nabla\varphi = \begin{pmatrix} \frac{\partial}{\partial x_1}\varphi \\ \frac{\partial}{\partial x_2}\varphi \end{pmatrix} = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}. \tag{14}$$

To determine the gradient of a triangle $\triangle = (i, j, k)$ with scalar values $\varphi_i := \varphi(\mathbf{p}_i)$, $\varphi_j := \varphi(\mathbf{p}_j)$, and $\varphi_k := \varphi(\mathbf{p}_k)$, a basis according to Equation 9 is built and the points $\mathbf{p}_i, \mathbf{p}_j, \mathbf{p}_k \in \mathbb{R}^3$ to $\mathbf{p}'_i, \mathbf{p}'_j, \mathbf{p}'_k \in \mathbb{R}^2$ are transformed by:

$$\mathbf{p}'_i = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \qquad \mathbf{p}'_j = \begin{pmatrix} \langle \mathbf{p}_j - \mathbf{p}_i, \mathbf{x}_\triangle \rangle \\ \langle \mathbf{p}_j - \mathbf{p}_i, \mathbf{y}_\triangle \rangle \end{pmatrix} \qquad \mathbf{p}'_k = \begin{pmatrix} \langle \mathbf{p}_k - \mathbf{p}_i, \mathbf{x}_\triangle \rangle \\ \langle \mathbf{p}_k - \mathbf{p}_i, \mathbf{y}_\triangle \rangle \end{pmatrix}.$$

This transformation describes an isometric (distance-preserving) and conformal (angle-preserving) map. The next step is a linearization of the scalar values $\varphi_i, \varphi_j, \varphi_k$. A scalar field $\varphi'(\mathbf{x}') = \alpha \cdot x'_1 + \beta \cdot x'_2 + \gamma$ is determined such that

$$\varphi'(\mathbf{p}'_i) = \varphi_i \qquad \varphi'(\mathbf{p}'_j) = \varphi_j \qquad \varphi'(\mathbf{p}'_k) = \varphi_k$$

holds. These conditions yield the following equation system:

$$\begin{pmatrix} \alpha & \beta \end{pmatrix} \begin{pmatrix} \mathbf{p}'_i & \mathbf{p}'_j & \mathbf{p}'_k \end{pmatrix} + \begin{pmatrix} \gamma & \gamma & \gamma \end{pmatrix} = \begin{pmatrix} \varphi_i & \varphi_j & \varphi_k \end{pmatrix}.$$

With $\mathbf{p}'_i = \binom{0}{0}$ the following solution is obtained:

$$\begin{aligned} \gamma &= \varphi_i, \\ \begin{pmatrix} \alpha & \beta \end{pmatrix} &= \begin{pmatrix} \varphi_j - \varphi_i & \varphi_k - \varphi_j \end{pmatrix} \begin{pmatrix} \mathbf{p}'_j & \mathbf{p}'_k \end{pmatrix}^{-1}. \end{aligned}$$

According to Equation 14, the gradient of φ' is determined by $\binom{\alpha}{\beta}$. The basis $\mathbf{x}_\triangle, \mathbf{y}_\triangle$ yields the gradient in 3D:

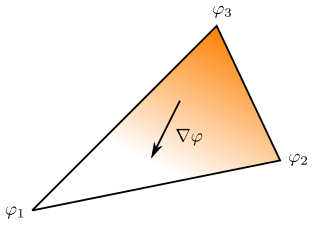


Figure 7: A triangle with different scalar values.

$$\nabla\varphi = \alpha \cdot \mathbf{x}_{\triangle} + \beta \cdot \mathbf{y}_{\triangle}. \tag{15}$$

Figure 7 illustrates the gradient of a triangle. To determine the gradient per vertex, the same procedure as for the shape operator estimation is used. The basis is transformed and the triangle gradients are weighted according to their proportion of the Voronoi area.

To determine the covariant derivative of a scalar field φ in direction of $\mathbf{v}$: $D_{\mathbf{v}}\varphi$, Equation 5 can be applied. Therefore, the gradient of φ is determined first and multiplied with $\mathbf{v}$ by using the dot product: $D_{\mathbf{v}}\varphi = \langle \nabla\varphi, \mathbf{v} \rangle$.

2.5.4 *Discrete Laplace-Beltrami Operator*

Several methods exist to discretize the Laplace-Beltrami operator on surface meshes for various applications. For an overview, the state of the art report by Sorkine [183] is recommended. The operator can be presented by the generalized formula:

$$\Delta\varphi(\mathbf{p}_i) = \sum_j w_{ij} \left(\varphi(\mathbf{p}_j) - \varphi(\mathbf{p}_i) \right).$$

Different weights w_j give different discrete Laplace-Beltrami operators. For presenting different versions of this operator, it is preferable that it fulfills some properties motivated by the smooth Laplace-Beltrami operator, recall Section 2.4.2:

(Sym) The weights should be symmetric $w_{ij} = w_{ji}$.

(Loc) If $(i, j) \notin E$ then $w_{ij} = 0$.

(Pos) All weights should be non-negative.

(Lin) If $\mathbf{p}_i$ is contained in a plane and φ is linear, then $\Delta\varphi(\mathbf{p}_i) = 0$ should hold.

In the following, different discrete Laplace-Beltrami operators will be introduced.

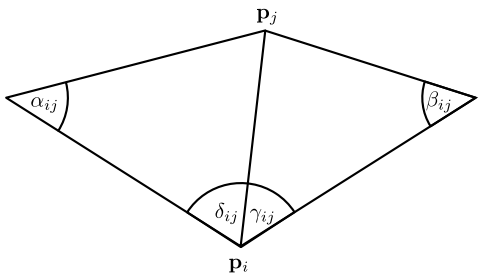


Figure 8: This figure illustrates the triangles with the angles for the weight calculation.

Combinatorial: The combinatorial Laplace-Beltrami operator is defined as:

$$w_{ij} = \begin{cases} 1, & \text{if } (i,j) \in E \\ 0, & \text{otherwise.} \end{cases}$$

This version may result in non-zero values on planar surfaces for linear scalar fields. Therefore, it violates (Lin).

Uniform: Taubin [193] suggested the uniform Laplace-Beltrami operator. The weights are determined by the number of neighbors of $\mathbf{p}_i$:

$$w_{ij} = \begin{cases} \frac{1}{N(i)}, & \text{if } (i,j) \in E \\ 0, & \text{otherwise.} \end{cases}$$

These weights also violate (Lin).

Floater s mean value: Floater [64] proposed the mean value weights by the tangent of the corresponding angles:

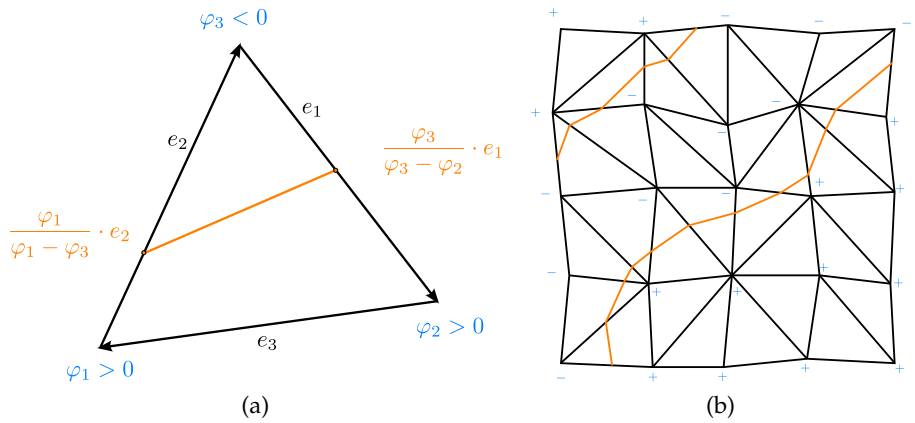
$$w_{ij} = \begin{cases} \frac{\tan(\delta_{ij}/2)+\tan(\gamma_{ij}/2)}{\|\mathbf{p}_i-\mathbf{p}_j\|}, & \text{if } (i,j) \in E \\ 0, & \text{otherwise.} \end{cases}$$

See Figure 8 for the angles. These weights violate (Sym).

Cotangent weights: MacNeal [133] suggested the cotangent weights:

$$w_{ij} = \begin{cases} \cot(\alpha_{ij}) + \cot(\beta_{ji}), & \text{if } (i,j) \in E \\ 0, & \text{otherwise.} \end{cases}$$

See Figure 8 for the angles. On general meshes, the weights can violate (Pos).



(a) (b)

Figure 9: In (a) the position of the zero crossing is determined and the points are connected. In (b) the isoline through a mesh is depicted.

Belkin weights: Belkin [12] suggested to determine weights over the whole surface:

$$\Delta\varphi(\mathbf{p}_i) = \frac{1}{4\pi h^2(\mathbf{p}_i)} \sum_{\triangle_k} \frac{A(\triangle_k)}{3} \sum_{j \in \triangle_k} e^{-\frac{\|\mathbf{p}_i - \mathbf{p}_j\|^2}{4h(\mathbf{p}_i)}} \Big(\varphi(\mathbf{p}_j) - \varphi(\mathbf{p}_i)\Big),$$

where $A(\triangle_k)$ denotes the area of the triangle $\triangle_k$ and h intuitively corresponds to the size of the neighborhood. This violates the (Loc) property.

Results: The discussion leads to the question if there is any discrete Laplace-Beltrami operator which fulfills all required properties for an arbitrary surface mesh. Wardetzky et al. [205] showed that there is no such operator. The proof is based on a Schönhardt polytope which demonstrates that no Laplace-Beltrami operator exists that fulfills all conditions.

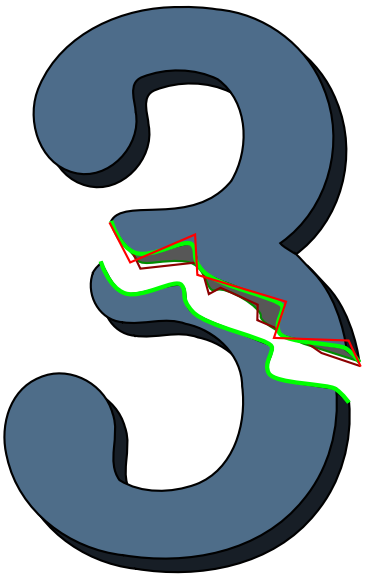
2.5.5 *Isolines on Discrete Surfaces*

For feature line methods, it is essential that the lines are not restricted to the edges, as it is not desirable to perceive the mesh edges. Given is a surface mesh and a scalar field, the zero crossings of the scalar field need to be depicted. Therefore, the scalar values for each triangle are linearized according to the values of the incident points. Afterwards, points along an edge are determined such that the linearized values of the scalar values of the connecting points equal zero. Two points on two edges of a triangle will be connected. Suppose a triangle with scalar values $\varphi_i > 0$, $\varphi_j > 0$ and $\varphi_k < 0$. Thus, somewhere on edge $\mathbf{e}_2$ and $\mathbf{e}_3$ is a zero crossing. The value $t = \frac{\varphi_k}{\varphi_k - \varphi_j}$ is determined and multiplied with the edge $\mathbf{e}_2$. This yields the position of the zero crossing on the edge. The position on the edge $\mathbf{e}_3$ is determined as well. Afterwards, both points will be connected, see Figure 9.

2.6 SUMMARY

This chapter introduced a special kind of illustrative visualization: line drawing. Here, feature lines and hatching were presented. Furthermore, the importance of illustrative visualization in the medical application was shown. Several examples were explained and referenced. Afterwards, the mathematical fundamentals were introduced. All relevant terms, which will be important for this thesis, were presented. Afterwards, the mathematical terms were adapted to the discrete field. Therefore, the continuous concept can be implemented on triangulated surface meshes, which is important for the further chapters.

Curve Smoothing on Triangulated Surfaces



This section is partly based on:

Kai Lawonn, Rocco Gasteiger, Christian Roessl
and Bernhard Preim
Adaptive and Robust Curve Smoothing on
Surface Meshes; Computer and Graphics,
40, pp. 22–35, 2014

Tobias Moench, Christoph Kubisch, Kai Lawonn
and Ruediger Westermann and Bernhard Preim
Visually Guided Mesh Smoothing for Medical Applications
Visual Computing for Biology and Medicine, pp. 91–98, 2012

Tobias Moench, Kai Lawonn, Christoph Kubisch
and Ruediger Westermann and Bernhard Preim
Interactive Mesh Smoothing for Medical Applications
Computer Graphics Forum, pp. 1–12, 2013

Mathias Neugebauer, Kai Lawonn, Oliver Beuing
and Bernhard Preim
Automatic generation of anatomic characteristics from
cerebral aneurysm surface models
Journal of Computer Assisted Radiology and Surgery,
pp. 279–289, 2013

3

CURVE SMOOTHING ON TRIANGULATED SURFACES

This chapter mainly presents an algorithm for the smoothing of curves on triangulated surface meshes. As this thesis focuses on vascular surfaces, an overview of the acquisition and surface reconstruction pipeline is given. The different acquisition steps are briefly described. Afterwards, the necessity of curve smoothing as well as cutting along such curves will be described.

3.1 ACQUISITION PIPELINE

This section briefly describes the steps to generate the vascular surface and blood flow data. The data acquisition pipeline consists of three steps: acquisition, surface reconstruction, and simulation.

3.1.1 *Acquisition*

In the first step, clinical image data (computed tomography angiography (CTA), magnetic resonance angiography (MRA), 3D rotational angiography) of the vessel are acquired. If phase-contrast magnetic resonance imaging (4D PC-MRI) is available, a full 3D flow measuring over time can be performed encoding the flow direction and magnitude at each voxel [134]. Measuring errors introduced by eddy currents, noise, and velocity phase wraps are reduced according to several filter methods described in Hennemuth et al. [82].

3.1.2 *Surface Reconstruction*

In the second step, the vessel surface is reconstructed based on a vessel segmentation. Because of the high vessel-to-tissue contrast in the image data, often a simple thresholding segmentation followed by a connected component analysis is sufficient to separate the vessel from the surrounding tissue. Artifacts, resulting from image acquisition and contrast agent distribution, can erroneously merge adjacent branches or lead to erroneous narrowings. They need to be manually corrected, see [144] for a description of this process. More advanced techniques like active shape models and deformable models can be employed to minimize the manual effort in cases of a low intensity distribution [125]. The resulting segmentation mask is used to reconstruct the surface with marching cubes and optimized with respect to

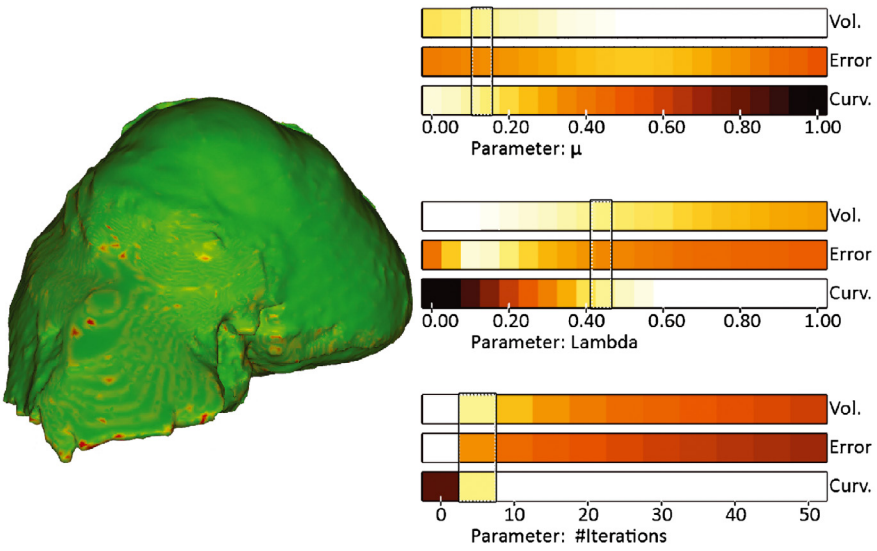


Figure 10: Result of a smoothed liver based on the parameters μ, lambda and iterations. (Images printed from Mönch et al. [228].)

mesh quality by a combination of edge collapses and edge flips [178].

3.1.2.1 *Surface Smoothing*

After the surface is reconstructed, the surface may exhibit artifacts like staircases, which need to be identified and reduced, see [143]. For the medical researcher it is important to get an opportunity to interactively smooth the surface model. Hence, one requirement is to use the tool without becoming acquainted with diverse smoothing algorithms. Therefore, knowing the influence of the parameters is essential for the understanding of the algorithm and to obtain appropriate results. This was the motivation of the work [227, 228]. Essentially, their work presents an interactive way to create appropriately smoothed surface meshes. During the smoothing process, the volume as well as the curvature may be reduced. A shrinking of the volume is a sign of losing the correctness in comparison to the original surface. Based on diverse graphs, e.g., curvature and volume, the medical researcher can smooth the model and is able to see the change of important attributes, see Figure 10. Furthermore, the system recommends optimal settings for a smoothed surface depending on related weights according to the original surface. If the user is interested in a smoothed mesh, which should maintain the volume according to the input mesh, the system proposes parameter settings for a reasonable result. Afterwards, the user may adjust the setting according to his individual tasks and preferences.

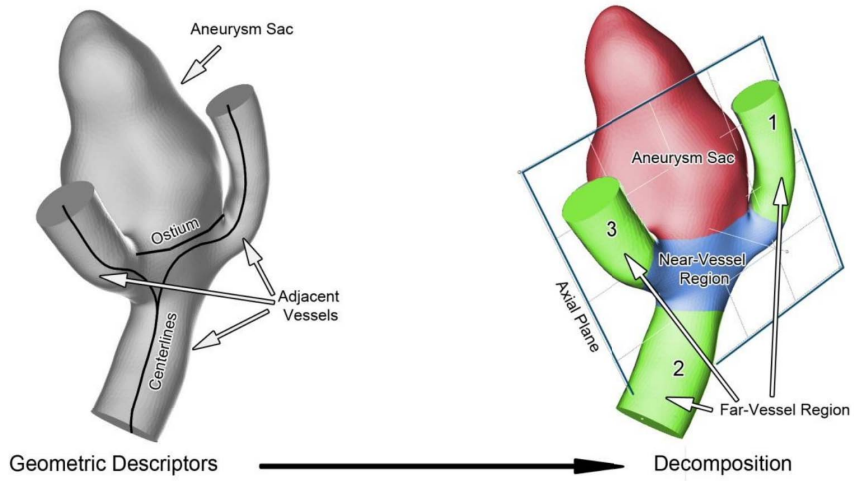


Figure 11: The vessel mesh, centerlines, and the ostium plane are used for decomposing the vessels into near- and far-vessel regions. (Images printed from Neugebauer et al. [230].)

3.1.2.2 *Identification of the Aneurysm*

For further evaluation of the vessel surface, an automatic approach to distinguish important from unimportant regions may be helpful. This is for example essential for the later CFD simulation. Having a decomposed vessel where the algorithm can identify the aneurysm and the branches, the CFD engineer can automatically set seed points for important structures and for analyzing essential flow patterns in interesting regions – the aneurysm. Furthermore, an automatic viewpoint selection helps the medical researcher to analyze and explore the vessel faster and helps him during the interaction. These requirements were the basis for the work [230]. The presented approach is able to decompose the adjacent vessels into near- and far-vessel regions, see Figure 11. By using the centerline of the vessel and the covariance matrix, the radius of the vessel at certain points on the centerline can be approximated. Afterwards, the Levenberg-Marquardt algorithm [145] is employed to fit a generalized Gaussian distribution to the radii of the vessel along the centerline. Characteristic points on the Gaussian distribution are used to identify the branches at the aneurysm. For the automatic viewpoint selection, an axial plane is calculated and the normal of the plane is used as the view vector for the camera. The determination of the automatic decomposition is very robust as it was tested on different data sets with different noise levels. The success of this approach was confirmed in an evaluation.

3.1.3 *Simulation*

The optimized surface mesh is utilized for generating an unstructured volume mesh, as an input for the subsequent CFD simulation.

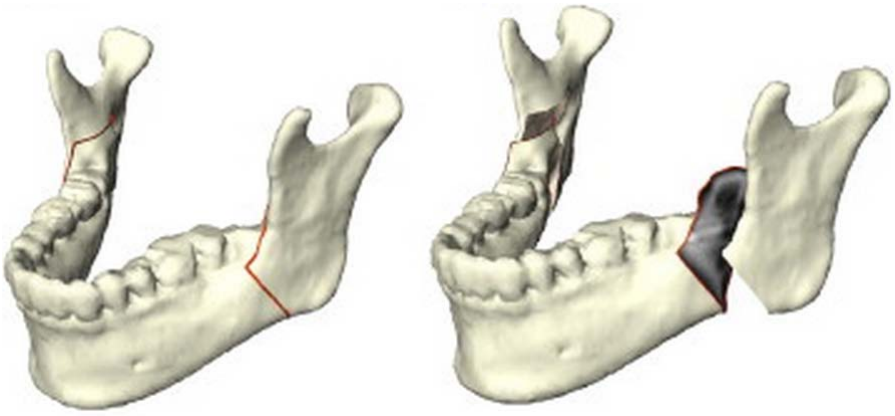


Figure 12: Sagittal split osteotomy along the red curve. (Images printed from Zachow et al. [210].)

In most cases, the blood is modeled as Newtonian fluid with steady or unsteady flow and rigid walls. Cebral et al. [34] have shown that with these assumptions a qualitative flow characterization is still possible.

3.2 EXTRACTION OF REGIONS

For general purposes, it is essential to distinguish important from unimportant structures. This may be applied for vessel surfaces where unimportant (long) vessels are cut. Then, the CFD simulation will be only employed for important vessels parts (including the aneurysm). Therefore, this section deals with placing a curve on the surface, smoothing it, and cut the structure along the newly generated curve.

3.2.1 *Introduction*

Curves on surfaces play an important role in many application domains, such as engineering or image-based medicine, where, e.g., surface cuts or segmentations are required [103, 210]. Especially in the field of surgery planning, surface cuts play an important role for discussing and planning. Osteotomies in maxillofacial surgery is an example where the cutting shows its potential [210], see Figure 12. One common requirement for such curves on surfaces is *smoothness*. This goal is typically achieved by smoothing an initial curve that might be roughly sketched by a user and that frequently shows noise.

For the Euclidean space $\mathbb{R}^2$, there are several methods for smoothing a given curve appropriately. In recent years, several approaches have been proposed that generalize these methods on 2D surface meshes and in Riemannian manifolds of arbitrary dimension. Given an initial curve, most of these approaches minimize certain energy functionals, such as the curve's length or its total curvature. Often,

the energy term is designed such that a curve aligns to the surface features. This arrangement leads to a curve that strongly deviates from the initial curve. For certain applications, such as treatment planning in surgery, this difference between the smoothed curve and the initial curve (where the initial curve was defined by a medical expert) should be small. Thus far, the existing solutions for this scenario suffer from a lack of convergence for noisy or irregularly tessellated surfaces. Furthermore, some of the existing approaches are limited to closed surface curves.

In this section a novel method for performing local smoothing of initial, jagged curves on triangulated surfaces based on iterative smoothing is proposed. This approach reduces a geodesic curvature while simultaneously controlling the deviation from the initial contour. The balance between smoothness and closeness is expressed by a single parameter. An additional parameter bounds the number of iterations, either directly or as an error threshold. The method ensures that the curve is always located on the surface, and the final result is comparable to different surface tessellations. Both requirements are necessary for applications involving surface cutting in medical applications, e.g., resection treatment planning in surgery. Experiments showed that this method is applicable to such scenarios with clinical data sets. It provides good results for synthetic benchmark surfaces.

In summary, the contributions of this work consist of an adaptive and novel approach to smooth surface curves that accomplishes the following:

- *preserves closeness* to the initial curve with respect to some bounding envelope,
- uses a *single parameter* to balance the closeness versus the smoothness with respect to the geodesic curvature,
- uses an abort criterion based on a theoretical *proof* for decreasing the curve's curvature, and
- is *robust* toward geometric and parametric noise.

3.2.2 *Related Work*

Smoothing surface curves is an important step for geometric processing, such as surface segmentation, editing, and cutting [13, 98, 210]. In most of these applications, an initial contour is defined either by direct user interaction or by (semi-)automatic feature detection. These initial contours are usually non-smooth and require a surface curve smoothing stage. This section focuses on this smoothing stage.

A very intuitive way to smooth a polygonal line is by corner cutting, which is well known as a *subdivision scheme* for planar curves

(see [36, 56]). Morera et al. [147] presented a generalization of subdivision algorithms for curves on surfaces. The resulting subdivision curve consists of points located anywhere in the interior of the surface triangles. Hence, the polyline that connects these points is not a surface curve (according to the definition of a surface curve in Section 3.2.4), and it might "miss" essential parts of the surface. Usually, approaches for smoothing surface curves should guarantee that the resulting curve is a part of the surface. For discrete surfaces that are represented as polygonal meshes, this arrangement means that there are curve samples on the *edges* of polygons such that every line segment of the curve is part of a surface polygon. Existing smoothing methods can be classified into methods that minimize energies or methods that approximate the given curve with (piecewise) polynomial curves.

ENERGY-MINIMIZING TECHNIQUES: Lee et al. [123, 124] combine automatic surface segmentation and cutting. After an initial feature contour is detected, a subsequent smoothing of the contour is performed by minimizing an energy-like functional. This functional is designed to meet different goals: to move the contour towards nearby features, to minimize the length of the contour, and to smooth the shape. Lai et al. [118] used a feature-sensitive curve smoothing for surface feature classification. They minimized a discretized tension spline energy with a subsequently projected gradient descent to obtain the smoothed boundaries. Kass et al. [105] represented curves as so-called *snakes*. A snake is a closed curve that evolves by minimizing internal forces, such as curvature, distances to features, or length, and external forces, such as distance to a feature. This approach is usually applied to image segmentation. Extensions to 2-manifold domains [13, 98, 16, 100] are successfully used to detect features on polygonal meshes. Most of these methods ensure an adaptive sampling of the snake depending on the mesh resolution. For snakes, a rapid movement towards the features is typically expected, and their initial shape is not important. Thus, none of these methods strives for closeness to the initial curve. In a different setting, Martínez et al. [136] focused on minimizing the local length of a given curve to obtain a geodesic on the surface. This approach is implemented by iteratively reducing the local length between two subsequent curve vertices. Similar to the previous methods, the resulting geodesic can differ strongly from the shape of the initial curve.

APPROXIMATION-BASED TECHNIQUES: There are several variants and extensions of the aforementioned approaches that achieve smoothing by "fitting" (piecewise) polynomial curves. Morera et al. [146] generalized Bézier curves in the Euclidean space $\mathbb{R}^2$ to geodesic Bézier curves on triangulated surfaces. The points on the initial curve

are used as control points for the geodesic Bézier curves located on the surface. Hofer and Pottmann [85] determined spline curves in manifolds by minimizing quadratic energies. Although most of the presented methods converge to smooth surface curves, they do not guarantee a closeness to the initial curve shape. Hofer and Pottmann [85] overcame this issue by adding more control points on the initial curve, but this approach requires more user interaction.

Another class of methods depends mainly on the surface features. These methods are designed to move the curve close to the features [13, 98, 16, 100]. However, depending on the field of application, both closeness and the independence of the surface features are important, i.e., the smoothing of the curve should not be related to the underlying surface features. This approach is inspired by the work of Martínez et al. [136] in terms of minimizing the geodesic curvature between the curve segments. In addition, this method allows for adjusting the closeness to the initial curve. The method neither depends on nor aligns to the surface features. It is largely independent of the particular surface tessellation and is robust against noise.

The approach smoothes the curves by reducing their geodesic curvature, i.e., the curve should evolve as straight as possible without unnecessary oscillations. In the limit, the algorithm obtains geodesics on the surface meshes.

There are several algorithms for computing such geodesics. These algorithms find a geodesic that connects two surface points, i.e., the algorithm solves a boundary value problem. In contrast, these algorithms either integrate a geodesic curve given a starting point and a direction (see Polthier and Schmies [161]), which means that they solved an initial value problem, or they computed *all* of the geodesics that emanate from a given vertex globally. The latter usually involves evolving fronts on the surface or the solution of a linear system. Mitchell et al. [142] presented an algorithm that finds the shortest path between two given points based on a continuous variant of Dijkstra's algorithm. Surazhsky and Surazhsky [187] extended this algorithm to obtain computationally efficient and accurate approximations. Therefore, they gained an exact solution more quickly. Bommes and Kobbelt [17] generalized [187] to handle arbitrary polygons on the mesh. Kimmel and Sethian [107] used the eikonal equation to generate a propagating front. The propagating front starts from a set of points and spreads over the mesh to calculate the distances from the start set. Recently, Crane et al. [44] proposed a method for computing geodesics using heat kernels on meshes. The gradient of the heat kernel is used to find an approximation of the eikonal equation. Therefore, the gradient of the heat kernel forms a new vector field. Afterwards, the new vector field is used to solve the Poisson equation and the resulting scalar field recovers the final distances.

3.2.3 *Motivation and Requirements*

The motivation of the approach is to achieve smooth surface curves from initial jagged curves for medical surface cutting applications. In contrast to non-medical applications, the smoothing requires a closeness to the initial curve shape, which is defined by domain experts such as physicians or bioengineers. In most cases, the initial curves indicate relevant anatomical landmarks or surface regions on which data analysis or treatment planning is performed. In particular, the algorithm focuses on the decomposition of vascular structures such as aneurysms for visual exploration purposes and on liver resections for preoperative treatment planning. For these applications, a patient-specific surface mesh is given, along with one or multiple user-defined cutting contours on the mesh, which represent a virtual resection or a decomposition for further analysis. In practice, the surface mesh is usually generated by the Marching Cubes algorithm and is based on a binary segmentation mask from medical image data, such as CT or MRI. The cutting contours are obtained by placing reference points on the mesh, which are connected by shortest path algorithms, such as breadth-first search, Dijkstra's algorithm [49] or similar approaches. The initial curves are continuous, and their segments are located entirely on the surface triangulation but suffer from a jagged curve shape. These noisy curves are distracting and would result in unpleasant surface cuts, which require more mental effort by the expert to conceive the cut shape. Moreover, the outlined anatomical landmarks and surface regions exhibit smooth shapes in reality but are approximated by the jagged curves. This arrangement can lead to inaccurate data analysis, such as area or volume estimation. Thus, an appropriate smoothing of these initial curves, which preserves the closeness to the initial shape is essential to support the visual perception and data analysis.

CHARACTERIZATION OF THE INPUT DATA. The curve smoothing algorithm operates on an arbitrary triangular surface mesh. For the clinical application of this work, surface extraction by Marching Cubes is highly efficient. However, the generated meshes often suffer from poorly shaped triangles. The resulting surface often contains noise: vertex distortions in the normal direction (geometric noise) and in a tangent space (parametric noise). These distortions are introduced due to beam hardening artifacts and noise in the image data. Furthermore, the binary segmentation mask can lead to block or staircase artifacts. Common approaches for image noise reduction, as well as binary mask and mesh smoothing, can reduce these artifacts. The degree of smoothing, however, must be carefully adjusted to prevent the elimination of relevant surface features and to preserve the volume [5]. There can also be "topological noise," such as small handles

or tunnels, which should be removed. Thus, some artifacts are still expected in the surface mesh. In practice, this circumstance also complicates or even hinders a (semi-)automatic remeshing to reduce the parametric noise, i.e., to improve the triangle quality. In summary, the initial situation is identified as follows: first, the given surface can show geometric, parametric and topological noise. Second, preprocessing of these data is not a viable option. Instead, algorithms are required that are robust enough to process these data.

This scenario is typical for medical research applications, such as investigations of simulated blood flow in cerebral aneurysms for rupture risk assessment [35]. Here, avoiding time-consuming and largely manual data preprocessing provides a significant benefit.

GOALS OF CURVE SMOOTHING. The overall goal is to construct a smooth curve based on the initial curve and the underlying surface mesh. By *smooth*, minimizing / reducing is referred to the initial geodesic curvature to obtain a surface curve that is "as straight as possible." However, the difference between the resulting smooth curve and the initial curve should be small because the initial curve is assumed to represent the region where the cut should occur. Given these two conflicting goals, several requirements must be fulfilled.

- *Adaptiveness:* The resolution of the evolving curve must adapt to the local mesh tessellation. This adaption requires a refinement and simplification of the curve during the smoothing steps.
- *Surface domain:* For the subsequent surface cutting, the smoothed curves must be located on the surface. Thus, the smoothing must be performed entirely on the surface, and every inserted or merged curve point must be located on the surface.
- *Robustness:* The smoothing must be robust with respect to both the surface artifacts and poor triangulation. In particular, any oscillating behavior must be avoided.

3.2.4 *Overview and Notation*

The proposed algorithm consists of three main steps.

1. **Initialization.** The user provides the initial curve, the desired curvature of the final curve, and its maximum distance to the initial curve.
2. **Smoothing.** This step computes weights for an iterative Laplacian *smoothing* of the curve. The specific choice of the weights ensures that all requirements are met. During the smoothing process, the curve points move along the surface edges within a user-defined region. This process could require a local adaptation: *splitting* and *merging* of the curve points.

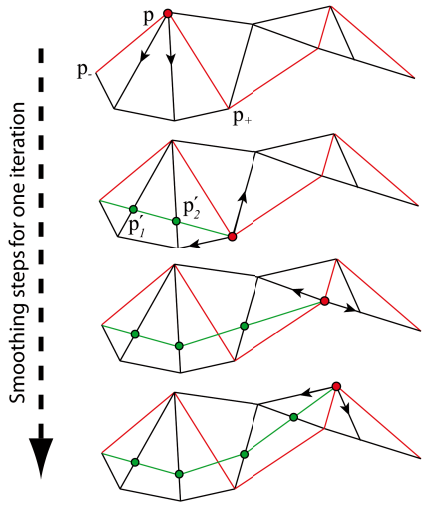


Figure 13: Illustration of three smoothing steps to shorten the initial curve (red). The points marked with a dot were moved along the edges to shorten the length between the predecessor p_- and the successor point p_+. In the next step, the previous successor point should be moved. After reaching the end of the curve, the next iteration starts.

3. **Evaluation.** Each smoothing step is followed by an evaluation of the curve to decide on the termination and to identify the critical surface vertices. Such vertices prevent a fast **movement**, and they need to be handled appropriately when the curve points are moving toward these vertices.

Steps 2 and 3 are iterated until the stopping criteria in step 3 are fulfilled. They assess the degree of smoothness and the distance from the initial contour. The general smoothing approach is illustrated in Figure 13, where one iteration with smoothing steps for three vertices is shown.

The following notation is used in the remainder of this section. Let $M \subset \mathbb{R}^3$ be a triangular mesh. The mesh consists of vertices V with associated positions $\mathbf{v}_i \in \mathbb{R}^3$, edges $E = \{(i,j) \,|\, i,j \in V\}$ and triangles $T = \{(i,j,k) \,|\, (i,j),(j,k),(k,i) \in E\}$. A surface curve C is defined as a sequence of points $\mathbf{p}_i \in \mathbb{R}^3$, which are connected by line segments and which lie on the surface mesh M. In particular, surface curves are considered, where for any segment i, its end points $\mathbf{p}_i$ and $\mathbf{p}_{i+1}$ are contained in two edges of a triangle $\Delta \in T$.

3.2.5 *Initialization*

The input of the algorithm consists of the initial curve and the desired curvature, together with the maximum distance to the initial curve. The initial curve is typically determined by the user interaction. The user adds points on the mesh that are then connected by the shortest

paths. If three adjacent points lie on one triangle, i.e., each point is associated with one vertex of that triangle, then the middle point is deleted to uniquely define the curve. The initial curve could be provided by other methods, such as feature-based segmentation. In the following, the curve is assumed to consist of a sequence of vertices that are connected by edges, i.e., $\mathbf{p}_i = \mathbf{v}_j, \mathbf{p}_{i+1} = \mathbf{v}_k$ with $(j,k) \in E$.

DESIRED CURVATURE Every point on the curve is assigned its initial geodesic curvature κ_g, with

$$\kappa_g = \pi - \frac{2\pi\beta}{\theta}$$

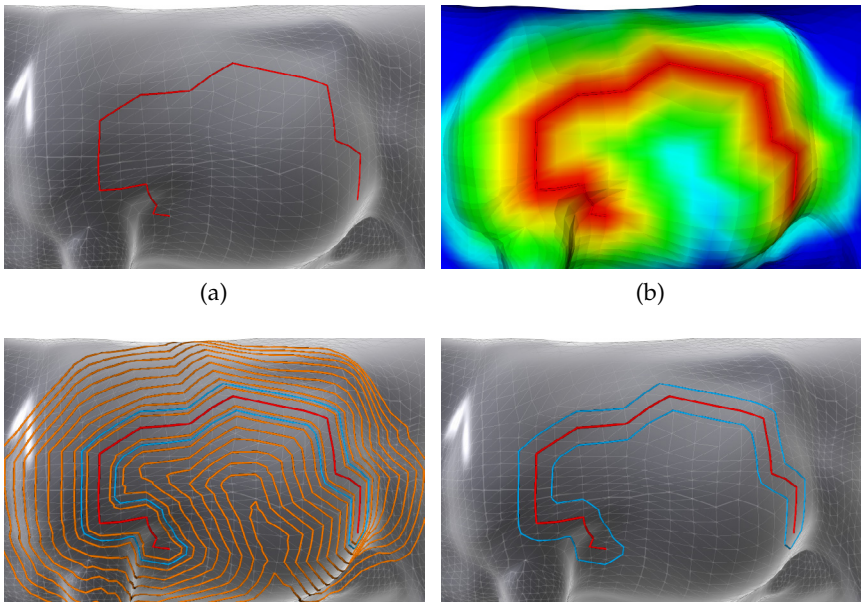
where θ is the total point angle, i.e., the sum of the internal angles of the adjacent triangles at the point, and β is one of the two curve angles. The angle β is chosen as the minimal angle of the two curve angles (see, e.g., [161]). If the curve intersects an edge, then $\theta = 2\pi$, which yields $\kappa_g = \pi - \beta$.
Additionally, every curve point $\mathbf{p}_i$ is assigned a *desired geodesic curvature* $\kappa_d(\mathbf{p}_i)$. The algorithm aims at moving curve points to positions where they have the approximate desired geodesic curvature. The desired geodesic curvature $\kappa_d(\mathbf{p}_i)$ is calculated by performing a linear interpolation between the geodesic curvature κ_g and 0 using a user-specified value t, as follows:

$$\kappa_d(\mathbf{p}_i) = t \cdot \kappa_g(\mathbf{p}_i), \quad t \in [0,1] \,. \tag{16}$$

Thus, for the value of $t = 1$, the algorithm should return the initial curve, and for $t = 0$, the smoothed curve represents the straightest geodesic. This arrangement means that the curve is as smooth as possible but can largely deviate from the initial curve. In this case, the algorithm obtains the same result as Martínez et al. [136].

MAXIMUM DISTANCES The movements of the curve points are restricted to *allowable regions* defined in terms of the Euclidean distance from the initial curve on the surface. If the initial curve points coincide with the vertex positions, then the allowable region T_{dist} is defined as the Euclidean distances of these vertices, which are less than dist, as illustrated in Figure 14. Formally, the distances can be determined for instances that have the fast marching approach or the geodesics in the heat approach [107, 44]. The domain experts can use a slider from 0 to the maximal distances from the initial curve on the whole surface. By interactively changing the values, the user obtains visual feedback via a contour line that depicts the corresponding distance from the curve. Therefore, the user can decide how far the smoothed curve is allowed to move.

For the experiments, an allowable region is determined by the 2-neighborhood of the vertices that coincide with the initial curve points.



(a) (b)

(c) (d)

Figure 14: Movement restriction: First, the initial curve is depicted in red (a). Afterwards, the geodesic distance is computed entirely on the mesh (b). Finally, the user can interactively change the distance. The result of the allowable region is depicted in cyan, and further contours are depicted in orange (c). The final allowable region is illustrated in (d).

If a straightest geodesic curve is desired, i.e., $t = 0$, then the curve should be allowed to move freely on the entire mesh.

3.2.6 *Curve Smoothing and Splitting Step*

The core part of the method is the iterative smoothing stage after initialization. In this process, the curve is relaxed such that curve points move but stay on edges. This process requires special treatment in the case where a curve point coincides with a vertex. The algorithm is constructed such that a curve point never moves across a vertex. This approach leads to two cases for the *smoothing* step for a single curve point $\mathbf{p}_i$:

Case 1: $\mathbf{p}_i$ is located on an edge.
Case 2: $\mathbf{p}_i$ is located on a vertex.

In the first case, $\mathbf{p}_i$ can be moved in two directions. Its destination is on the edge or on one of the vertices that span the edge. The second case requires a *splitting*: as the point moves away from the vertex, new curve segments are required. Thus, $\mathbf{p}_i$ must be split into multiple curve points, each of which is located on edges incident to the vertex. The two cases for a single point relaxation will be described

and analyzed. One curve smoothing step in the iteration consists of the relaxation of all of the curve points.

3.2.6.1 *Case 1:* $\mathbf{p}_i$ *is located on an edge*

Let $\mathbf{p} := \mathbf{p}_i \in C$ denote a curve point on an edge e, and let $\mathbf{p}_- := \mathbf{p}_{i-1}$ and $\mathbf{p}_+ := \mathbf{p}_{i+1}$ denote its neighbors. If $\mathbf{p}$ is placed on an edge $e \in E$, then a Laplacian smoothing $\mathcal{L}_e(\mathbf{p})$ is applied that determines a new position of $\mathbf{p}$ as a linear combination of $\mathbf{p}_-$ and $\mathbf{p}_+$:

$$\mathcal{L}_e(\mathbf{p}) \;=\; \mathbf{p} \;+\; \big(\lambda \cdot \omega_- \;\mathrm{proj}_e(\mathbf{p}_- - \mathbf{p}) \;+\; \lambda \cdot \omega_+ \;\mathrm{proj}_e(\mathbf{p}_+ - \mathbf{p})\big)\,.$$

In contrast to a standard Laplacian relaxation, the linear combination *weights* $\omega_\pm$ with the *correction factor* λ are constructed such that $\mathcal{L}_e(\mathbf{p})$ keeps $\mathbf{p}$ on the edge e. The operator proj_e projects points onto the line spanned by e. Furthermore, $\omega_\pm \geqslant 0$ and $\omega_+ + \omega_- = 1$ will be postulated. This arrangement leads to the following:

$$\mathcal{L}_e(\mathbf{p}) = (1-\lambda)\cdot\mathbf{p} + \lambda\cdot\big(\omega_-\;\mathrm{proj}_e(\mathbf{p}_-) + \omega_+\;\mathrm{proj}_e(\mathbf{p}_+)\big). \quad (17)$$

The crucial part remains the definition of the *weights* $\omega_\pm$ and the *correction factor* λ such that convergence to the desired curvature is achieved.

COMPUTATION OF THE WEIGHTS The curve point $\mathbf{p}$ is located on the edge e. Let $\mathbf{v}_1$ and $\mathbf{v}_2$ denote the vertices that span e. The weights are defined as

$$\omega_- = \frac{\mathrm{dist}\,\mathbf{p}_+\mathrm{proj}_e(\mathbf{p}_+)}{\mathrm{dist}\,\mathbf{p}_+\mathrm{proj}_e(\mathbf{p}_+) + \mathrm{dist}\,\mathbf{p}_-\mathrm{proj}_e(\mathbf{p}_-)}$$

and

$$\omega_+ = \frac{\mathrm{dist}\,\mathbf{p}_-\mathrm{proj}_e(\mathbf{p}_-)}{\mathrm{dist}\,\mathbf{p}_+\mathrm{proj}_e(\mathbf{p}_+) + \mathrm{dist}\,\mathbf{p}_-\mathrm{proj}_e(\mathbf{p}_-)}\,,$$

where $\mathrm{dist}\,\cdot\cdot$ denotes the Euclidean distance.

First, it will be proven that setting $\lambda = 1$ yields a shortest path that connect $\mathbf{p}_-$ and $\mathbf{p}_+$ via a point on the edge.

Theorem 3.2.1. *Let* $\lambda = 1$. *Then, the partial curve spanned by the sequence* $[\mathbf{p}_-, \mathcal{L}_e(\mathbf{p}), \mathbf{p}_+]$ *is a geodesic in* M.

Proof. Let $\mathbf{p}^*$ be the point on edge e such that the distance between $\mathbf{p}_-$ and $\mathbf{p}_+$ via $\mathbf{p}^*$ is minimal, i.e., $[\mathbf{p}_-, \mathbf{p}^*, \mathbf{p}_+]$ is a geodesic curve in M. The general case that $\mathbf{p}_-, \mathbf{p}_+, \mathbf{v}_1$ and $\mathbf{v}_2$ are not collinear is assumed, which would degenerate to the trivial case. The problem is simplified by two rigid transformations: first, $\mathbf{p}_-$ is shifted to the origin. Then, the configuration is rotated such that the linear segment $[\mathbf{v}_1, \mathbf{v}_2]$, which spans e, is located in the xy-plane and is perpendicular to the x-axis. Finally, the point $\mathbf{p}_+$ is rotated *only* around the edge

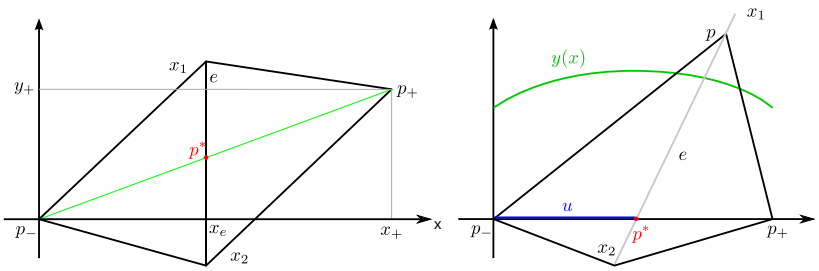


Figure 15: The problem is transformed in the xy-plane and search for the point p^*.

e such that $\mathbf{p}_+$ lies in the xy-plane. The first translation and the rotation of the whole configuration preserve the lengths. The last rotation unfolds the two neighboring triangles at e into the xy-plane, i.e., it preserves the lengths as measured on the surface mesh M. Figure 15a illustrates the situation.

It will be shown that $\mathbf{p}^* = \mathcal{L}_e(\mathbf{p})$, and it is sufficient to do this step in $\mathbb{R}^2$ after transformation into the xy-plane with $\mathbf{p}_- = (0,0)^T$, $\mathbf{p}_+ = (x_+, y_+)^T$, $\mathbf{v}_1 = (x_e, y_1)^T$, and $\mathbf{v}_2 = (x_e, y_2)^T$. The line through $\mathbf{p}_-$ and $\mathbf{p}_+$ is given as $x_+ y = y_+ x$ (with $x_+ \neq 0$ for the nontrivial case. The point $\mathbf{p}^*$ is obtained as the intersection of the edge and the straight line through $\mathbf{p}_-$ and $\mathbf{p}_+$ with $\mathbf{p}^* = (x_e, \frac{y_+}{x_+} x_e)$. It remains to show the equality $\mathcal{L}_e(\mathbf{p}) = \mathbf{p}^*$. Using Eq. 17 with $\lambda = 1$ yields:

$$
\begin{aligned}
\mathcal{L}_e(\mathbf{p}) = & \frac{\operatorname{dist} p_+ \operatorname{proj}_e(p_+)}{\operatorname{dist} p_+ \operatorname{proj}_e(p_+) + \operatorname{dist} p_- \operatorname{proj}_e(p_-)} \cdot \operatorname{proj}_e(p_-) \\
& + \frac{\operatorname{dist} p_- \operatorname{proj}_e(p_-)}{\operatorname{dist} p_+ \operatorname{proj}_e(p_+) + \operatorname{dist} p_- \operatorname{proj}_e(p_-)} \cdot \operatorname{proj}_e(p_+) \\
= & \frac{x_+ - x_e}{x_+ - x_e + x_e} \operatorname{proj}_e(p_-) + \frac{x_e}{x_+ - x_e + x_e} \operatorname{proj}_e(p_+) \\
= & \frac{x_+ - x_e}{x_+} \cdot \begin{pmatrix} x_e \\ 0 \end{pmatrix} + \frac{x_e}{x_+} \cdot \begin{pmatrix} x_e \\ y_+ \end{pmatrix} = \begin{pmatrix} x_e \\ x_e \frac{y_+}{x_+} \end{pmatrix} = p^*,
\end{aligned}
$$

which proves Theorem 3.2.1. □

COMPUTATION OF THE CORRECTION FACTOR The correction factor λ ensures that a point on the curve will evolve on the surface such that the desired geodesic curvature is achieved. To determine λ, the two neighboring triangles will be unfolded in the plane as before. However, the planar configuration is rotated such that the line ℓ through connecting points $\mathbf{p}_-$ and $\mathbf{p}_+$ coincides with the x-axis (see Figure 15b). The point $\mathcal{L}_e(\mathbf{p})$ along the edge e is spanned by $\mathbf{v}_1$ and $\mathbf{v}_2$.

First, Thales' theorem will be generalized. Given the points $\mathbf{p}_-, \mathbf{p}_+$, the function $(x, y(x))$ needs to be determined such that the enclosed

angle $\sphericalangle(\mathbf{p}_-, (x, y(x)), \mathbf{p}_+) = \vartheta$. Here, ϑ is the converted angle from κ_d, as described in [161]. The assignment $T := \tan(\vartheta - \pi)$ and $L := \|\mathbf{p}_+ - \mathbf{p}_-\|$ yields:

$$y(x) = \frac{L}{2T} + \sqrt{\frac{L^2}{4T^2} + x\,(L - x)}. \tag{18}$$

This statement can be proven by using the definition of tan and by applying trigonometric identities:

$$\vartheta = \arctan\frac{x}{y} + \arctan\frac{L-x}{y} = \pi + \arctan\frac{\frac{x}{y} + \frac{L-x}{y}}{1 - \frac{x(L-x)}{y^2}}$$

$$\tan(\vartheta - \pi) = \frac{yL}{y^2 - x(L-x)}.$$

Next, the position $\mathbf{p}^{*\prime}$ is determined on the edge e, where

$$\sphericalangle(\mathbf{p}_-, \mathbf{p}^{*\prime}, \mathbf{p}_x) = \vartheta.$$

Thus, the intersection point of $(x, y(x))$ and the span of the edge e must be calculated. Edge e intersects the x-axis at an angle $\gamma < \frac{\pi}{2}$ in such away that the span can be written as $y(x) = \tan(\gamma)(x - u)$ or

$$x = \frac{y}{\tan\gamma} + u\,. \tag{19}$$

Inserting (19) into (18) leads to

$$y = \frac{L}{2T} + \sqrt{\frac{L^2}{4T^2} + \left(\frac{y}{\tan\gamma} + u\right)\left(L - \frac{y}{\tan\gamma} - u\right)}\,.$$

Further simplification yields

$$0 = y^2 \underbrace{\left(1 + \frac{1}{\tan\gamma^2}\right)}_{=:\alpha} - y \underbrace{\left(\frac{L}{T} + \frac{L - 2u}{\tan\gamma}\right)}_{=:\beta} - u(L-u)\,.$$

A positive solution for y is obtained as

$$y^* = \frac{\beta}{2\alpha} + \sqrt{\frac{\beta^2}{4\alpha^2} + \frac{u(L-u)}{\alpha}}\,.$$

The altitude of the triangle is calculated to determine λ:

$$d = \frac{\|(\mathbf{p} - \mathbf{p}_-) \times (\mathbf{p}_+ - \mathbf{p}_-)\|}{\|\mathbf{p}_+ - \mathbf{p}_-\|}$$

and set the correction factor λ to:

$$\lambda = 1 - \frac{y^*}{d}\,.$$

Then, for $\vartheta \to \pi$, the curve is locally a straightest geodesic with $\lambda \to 1$, and Theorem 3.2.1 applies. Hence, the smoothing operator $\mathcal{L}_e$ with the weights ω_- and ω_+ and the correction factor λ defined as above yields the straightest geodesic curves.

3.2.6.2 *Case 2: $\mathbf{p}_i$ is located on a surface vertex*

The second case applies if the curve point $\mathbf{p}$ is located on a surface vertex $i \in V$, i.e., $\mathbf{p} = \mathbf{v}_i$. In this case, plain relaxation as described in Section 3.2.6.1 is not sufficient. Instead, the curve must be locally split into multiple segments, which are spanned between the edges in $\text{star}(i) := \{(i,j) \in E \,|\, j \in V\}$. Given the curve segment $[\mathbf{p}_-, \mathbf{p}, \mathbf{p}_+]$, the neighbor vertices j of i in $\text{star}(i)$ is partitioned into two sequences. The first sequence N_1 enumerates the neighbors counterclockwise, starting from the edge of $\mathbf{p}_-$ and ending on the edge of $\mathbf{p}_+$. The second sequence N_2 enumerates the remaining neighbors clockwise around vertex i. Let $\mathbf{p}^1_k$ and $\mathbf{p}^2_k$ denote the positions of the vertices in N_1 and N_2, respectively, with the additional end points $\mathbf{p}^1_0 = \mathbf{p}^2_0 = \mathbf{p}_-$ and $\mathbf{p}^1_m = \mathbf{p}^2_n = \mathbf{p}_+$.

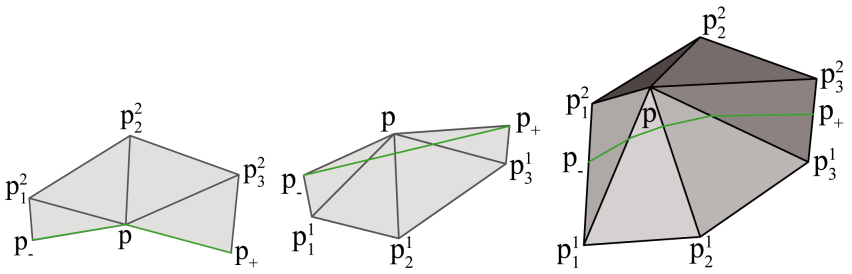
SPLITTING After local relaxation, $\mathbf{p}$ will be replaced by a new sequence of points located on the edges $E_1 = \{(i,j) \,|\, j \in N_1\}$ or $E_2 = \{(i,j) \,|\, j \in N_2\}$. The decision between the two options is based on a local parameterization of $\mathbf{p}^1_k$ and $\mathbf{p}^2_k$: the local surface patch is cut along $[\mathbf{p}_-, \mathbf{p}, \mathbf{p}_+]$ and unfold the two parts using an exponential map. The exponential map is a local map from the tangential space around a vertex to the mesh. Around a small neighborhood, this map is a diffeomorphism. Therefore, this definition is used to unfold the star of a vertex on the corresponding tangent space (see, e.g., [161]).

SMOOTHING Based on the configuration described above, the new curve points are computed between $\mathbf{p}^c_0$ and $\mathbf{p}_+$ in the split curve segment as follows. The algorithm turns counterclockwise ($\mathbf{p}^1_k$) and clockwise ($\mathbf{p}^2_k$) around the vertex at $\mathbf{p} = \mathbf{v}_i$. Starting with $k = 0$, the pairs $\mathbf{p}^c_k, \mathbf{p}^c_{k+1}$ are considered to determine the position of a new curve point $\mathbf{p}'$ on the line segment $[\mathbf{v}_i, \mathbf{p}^c_{k+1}]$, as described in Section 3.2.6.1. Then, $\mathbf{p}^c_{k+1} \leftarrow \mathbf{p}'$ is set and proceed with $k \leftarrow k+1$ until $\mathbf{p}^c_k = \mathbf{p}_-$ is reached. Figure 16 illustrates this procedure. Invalid configurations can arise in this process whenever a line segment is not contained in the unfolded parameter domain. In this case, the line segment is replaced by parts of the domain boundary.

Finally, the lengths of the two curve segments $\mathbf{p}^1_k$ and $\mathbf{p}^2_k$ are compared after splitting and smoothing, and the algorithm chooses the shorter one to replace the original segment $[\mathbf{p}_-, \mathbf{p}, \mathbf{p}_+]$. For this result, the following theorem will be proven.

Theorem 3.2.2. *Let $\lambda = 1$. Performing splitting and smoothing as described above computes the shortest surface curve connecting $\mathbf{p}_-$ and $\mathbf{p}_+$.*

Proof. For each patch N_1 and N_2 unfolded in the plane, the curve points are founded on the surface edges that minimize the distances locally for each segment of the split curve if the sum of the inner angles is less than π (Theorem 3.2.1). The first line from $\mathbf{p}_-$ to $\mathbf{p}_+$



(a) N_1 (b) N_2 (c)

Figure 16: Locally shortest paths after cutting and unfolding the neighbors N_1, N_2 of the domain in (c) restricted to the local domains (a) and (b). The shorter curve is selected.

over $\mathbf{p}^{c\prime}$ on the line segment $[\mathbf{v}_i, \mathbf{p}_1^c]$ determines the remaining points on the edges $(\mathbf{p}, \mathbf{p}_2^c), \ldots (\mathbf{p}, \mathbf{p}_{|N_c|-1}^c)$. Next, the intersection point of the line and the remaining edges must be calculated. This approach minimizes the distance between $\mathbf{p}_-$ and $\mathbf{p}_+$ because the shortest connection on a plane is a straight line. The parameterization preserves the length of the line on the surface and the angle between the line and its edges. Because the parameterization unfolds the fan in the plane and the algorithm can find the shortest line from $\mathbf{p}_-$ to $\mathbf{p}_+$, the intersection points are used as candidates for the points that lie on the edges of the neighboring set N_c. Afterward, the resulting points are transformed back to the surface. Furthermore, if the connection line between $\mathbf{p}_-$ and $\mathbf{p}_+$ is a non-valid line, i.e., some line parts are outside the triangle fan, it will be replaced with the shortest line between $\mathbf{p}_-$ and $\mathbf{p}_+$. □

3.2.6.3 *Convergence of the Algorithm*

The goal is to ensure that the algorithm terminates. If there is no number of iterations given, then the constraints are defined when a point is allowed to move. These rules ensure convergence of the curvature. First, it will be shown that the length of the curve decreases for $\lambda = 1$. This approach guarantees that the length converges during the iteration. Then, for the case $\lambda \neq 1$, two constraints are defined when a point is allowed to move. This strategy guarantees that the curvature converges during the iteration. Finally, the same abort criterion for $\lambda = 1$ and for $\lambda \neq 1$ is defined.

Theorem 3.2.3. *Let $\lambda = 1$. The length of the curve decreases for every iteration step.*

Proof. In every iteration step, a virtual line sweeps over each curve point and reduces the distance between its predecessor and successor point. Thus, instead of proving that every iteration step reduces the

length of the curve from the previous iteration, it will be shown that the sweep line shortens the length of the curve in each iteration step. For the first sweep line position, the line is placed at a curve point $\mathbf{p}$. The points $\mathbf{p}_-, \mathbf{p}_+$ are kept and move $\mathbf{p}$ to $\mathbf{p}'$ or to $\mathbf{p}'_1, \mathbf{p}'_2, \ldots, \mathbf{p}'_n$, depending on whether $\mathbf{p}$ lies on an edge or on a vertex in which the distance of $\mathbf{p}_-$ to $\mathbf{p}_+$ via $\mathbf{p}'$ or $\mathbf{p}'_1, \mathbf{p}'_2, \ldots, \mathbf{p}'_n$, respectively, is minimal. The next sweep line position is the point $\mathbf{p}_+$, which is moved to its relative optimal position in terms of its predecessor and successor (recall Figure 13). As the sweep line reduces the length of the curve in the first sweep, it also reduces the length in the next sweep step. When the sweep line reaches the end, the length of the curve is decreased, and thus, the length is reduced after each iteration step. □

For $\lambda \neq 1$, the curve is restricted to change only if the geodesic curvature decreases in every iteration step. To ensure this aspect, two different constraints are applied to the smoothing algorithm. First, whenever the current geodesic curvature $\kappa_{cur}(\mathbf{p})$ of a certain point $\mathbf{p}$ is less than or equal to the desired geodesic curvature $\kappa_d(\mathbf{p})$ of this point

$$\kappa_{cur}(\mathbf{p}) \leqslant \kappa_d(\mathbf{p}), \tag{20}$$

it does not allow the point to move. Second, a point $\mathbf{p}$ is allowed to move only if the sum of the geodesic curvature of this point and its predecessor and successor point after the movement is less than or equal to the sum of the geodesic curvatures κ' before movement

$$\kappa(\mathbf{p}) + \kappa(\mathbf{p}_+) + \kappa(\mathbf{p}_-) \leqslant \kappa'(\mathbf{p}) + \kappa'(\mathbf{p}_+) + \kappa'(\mathbf{p}_-). \tag{21}$$

Therefore, the sum of the geodesic curvatures decreases for every iteration step. The proof is similar to the proof of Theorem 3.2.3. Both properties are used as an abort criterion for the smoothing algorithm. For $\lambda \neq 1$ ($t \neq 0$), it will be shown that the sum of the geodesic curvatures decreases for every iteration step. Furthermore, $\sum \kappa \geqslant 0$ is a bounded and monotonic series, which means that it converges. Thus, the abort criterion is defined in such a way that the smoothing process stops if the change in the geodesic curvature from one iteration step to the next is not significant. The case $\lambda = 1$ leads to a curve shortening flow. Recent work addresses the convergence of this flow with closed initial curves. Ma and Chen [132] showed that if the shortening flow exists for a finite amount of time on a compact Riemannian manifold and the limit of the length of the curve is greater than zero, the limiting curve exists and is a geodesic. Furthermore, one characteristic is that the derivative of the curve's length is equal to the negative integral over the squared curvature:

$$\frac{d}{dt}L = -\int \kappa^2 \, ds.$$

The length also converges because it is a bounded monotonic series; thus, the derivative becomes zero for infinite time steps, and the curvature goes to zero as well. Therefore, the abort criterion can be used for $\lambda = 1$ as well as for $\lambda \neq 1$.

3.2.7 *Curve Evaluation*

After each smoothing iteration, the curve is evaluated to test for the stopping criteria and critical vertex configurations. Critical vertices prevent fast smoothing and could lead to sharp edges. Thus, they must be identified and handled by merging.

Abort Criteria: Two criteria are used to stop the smoothing process. First, the smoothed curve should not move too far from the initial curve. This constraint is ensured by restricting the movement of the curve points to the allowable envelope defined in Section 3.2.5. Whenever a point would move out of this region, the iteration is stopped. Second, the iteration stops if the curve has converged to the prescribed desired geodesic curvature κ_d. A new tolerance parameter τ is defined to relax the interpolation of the desired geodesic curve. If every current geodesic curvature deviates from the desired geodesic curvature by **less** than the defined τ-percentage, then this iteration stops.

Critical Vertices: Critical vertices are "surrounded" by curve points; that converge to these vertices but do not *cross* them (see Figure 17). This arrangement means that these vertices prevent fast convergence of the smoothing, and they must be treated specially.
A vertex $i \in V$ is *potentially* a critical vertex if there are curve segments with the associated curve points $\mathbf{p}_i, \mathbf{p}_{i+1}, \ldots, \mathbf{p}_{i+\ell}$, where a subsequence $\mathbf{p}_{i+1}, \ldots, \mathbf{p}_{i+\ell-1}$ is located on the edges $e_{i+1}, \ldots, e_{i+\ell-1}$ connected to vertex i, *and* the points $\mathbf{p}_i, \mathbf{p}_{i+\ell}$ are located on edges that do not contain the vertex i. After each iteration step, the algorithm identifies these candidates and "simulate" the following merge operation: the curve points $\mathbf{p}_{i+1}, \ldots, \mathbf{p}_{i+\ell-1}$ are merged to a single point located at the vertex position $\mathbf{v}_i$. The algorithm tests if the ratio of the lengths of the curve after and before merging exceeds a threshold ϵ. In this case, the algorithm applies the merge operation. Otherwise, the algorithm continues with the original curve. If the user wants to find the straightest geodesic, i.e., the parameter $t = 0$ (see Sec. 3.2.5), then ϵ is omitted. This circumstance means that merging is applied whenever the new length becomes shorter. For $t \neq 0$, a value of $\epsilon = 0.98$ is suggested based on empirical observations. This arrangement ensures a merging whenever the curve length from $\mathbf{p}_{i+1}$ to $\mathbf{p}_{i+l-1}$ is at least ten times shorter than the minimal distance from $\mathbf{p}_i$ to $\mathbf{p}_{i+1}$ or from $\mathbf{p}_{i+l-1}$ to $\mathbf{p}_{i+l}$. After merging the points $\mathbf{p}_{i+1}, \ldots, \mathbf{p}_{i+\ell-1}$ to the vertex position $\mathbf{v}_i$, the algorithm uses the median of the desired geodesic curvature of the points and assign it to the new point.

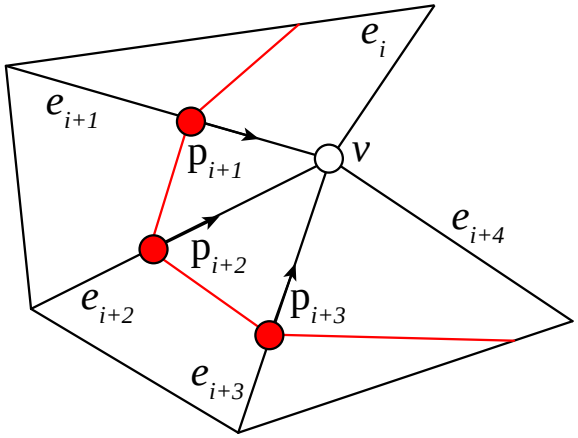


Figure 17: Critical vertex: The red curve converges towards $\mathbf{v}$ but does not "cross" the vertex.

3.2.8 *Algorithm*

The algorithm can be summarized with the pseudo-code shown in Listing 2. The functions `initialize()`,

Algorithm 2 Pseudo-code for the algorithm.

```
initialize(p)
[φ]=computeDesiredAngles(p)
T=computeAllowableRegion(p)
for smoothing iterations
  for all curve points p_i
     [N_1,N_2]=getNeighborSets(p_i)
     selectNeighbor(p_i,N_1,N_2)
     p_i=relocatePoint(p_i,φ_i,[N_1,N_2])
  end
  if (testAbortCriteria(p,T)) then break; end
  handleCriticalVertices()
end
```

`computeDesiredAngles()`, and `computeAllowableRegion()` create the input curve and compute the desired geodesic curvature and a maximum distance-based feasible region, as described in Sec. 3.2.5. Then, the algorithm start the smoothing iteration. In every iteration, the algorithm applies the following for every point $\mathbf{p}_i$: Here, the function `getNeighborSets()` computes the sets N_1 and N_2, and `selectNeighbor()` selects the appropriate set depending on the unfolded configuration. Finally, `relocatePoint()` evaluates the new position of $\mathbf{p}_i$ (see Sec. 3.2.6). After each iteration, the function `testAbortCriteria()` is evaluated, which eventually terminates the algorithm. Special cases are treated in the last function `handleCriticalVertices()` (Sec. 3.2.7).

The presentation will be closed with a few remarks. For simplicity, only the distances of the points on the current curve to their corresponding original points are compared. If a point is split, then the k-neighborhood distance of that point is assigned to the newly inserted points for the allowable region test. Similarly, if the points are merged, then the maximum k-neighborhood distance of all of the merged points is assigned to the new point. Finally, the points are relocated only if the current geodesic curvature is greater than the desired geodesic curvature and if the current geodesic curvature deviates from the desired geodesic curvature by more than τ-percentage (recall Sec. 3.2.7). The experiments demonstrate that $\tau = 10\%$ is a reasonable value.

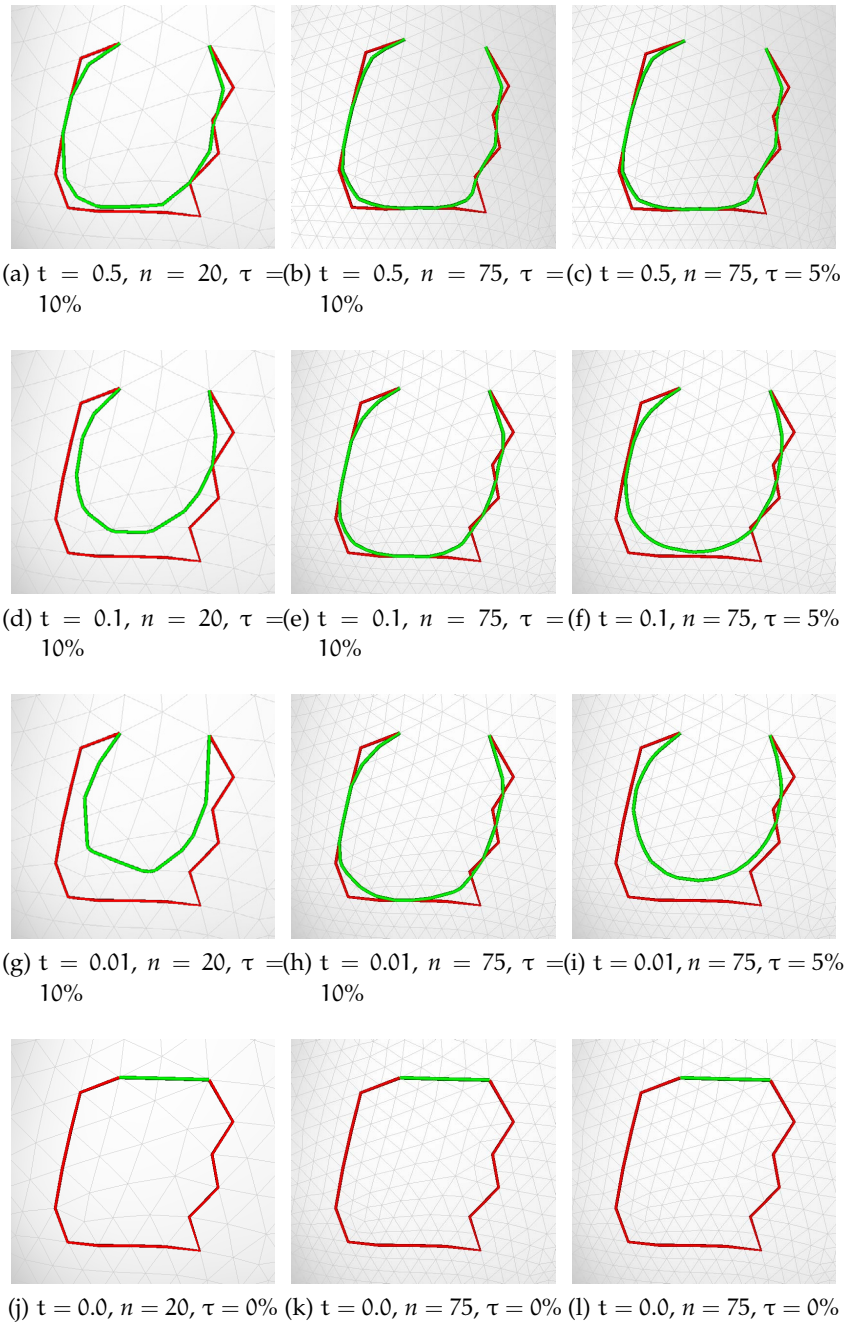
3.2.9 *Results and Application*

The method was evaluated on artificial and real-world surface data sets to verify its robustness and convergence. By convergence, it is meant that the condition $t \approx \frac{\kappa'_g}{\kappa_g}$ is fulfilled, with κ'_g being the geodesic curvature after smoothing. The real-world data are anatomical surfaces that are patient-specific and representative for two medical applications: vascular models of cerebral aneurysms for decomposition and liver models for resection planning. All of the tests are performed on an Intel Core 2 Duo CPU at 3.16GHz. The memory requirements for the curve smoothing are negligible compared to the memory required by the data sets.

To specify an initial curve, the user selects a sequence of vertices connected by shortest edge paths using Dijkstra's algorithm. Furthermore, the user specifies the parameter t, which defines the globally desired geodesic curvature. Optionally, the user can relax this specification by varying the tolerance parameter τ (Sec. 5.3). For all of the experiments, $\tau = 10$ % is used unless otherwise specified.

3.2.9.1 *Convergence and Robustness*

Two experiments were performed to assess the convergence and robustness of the approach. For the convergence, the smoothing for different parameter settings and mesh resolutions was investigated. Figure 18 shows the results for varying the parameters $t \in \{0.5, 0.1, 0.01, 0\}$ and $\tau \in \{10\%, 5\%, 0\%\}$ after a fixed number of iterations $n \in \{20, 75\}$. It was observed that, as expected for a decrease in t, the curve changes gradually from the initial curve to the straightest geodesic. In the second column of Figure 18, each triangle was subdivided into four triangles. In this case, the number of iterations was increased from 20 to 75. It was observed that for $\tau = 10\%$, the curve converges to a smooth curve close to the original unless $t = \tau = 0$ is set, for which the curve converges to a straightest geodesic. The third column of



(a) $t = 0.5$, $n = 20$, $\tau = 10\%$ (b) $t = 0.5$, $n = 75$, $\tau = 10\%$ (c) $t = 0.5$, $n = 75$, $\tau = 5\%$

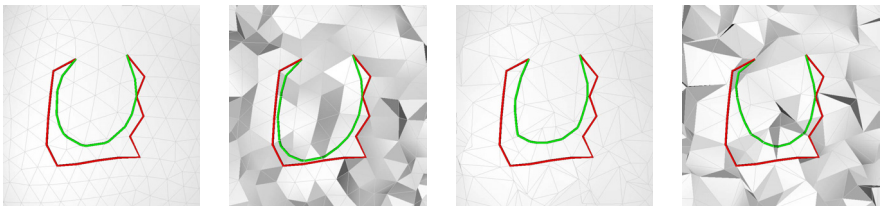
(d) $t = 0.1$, $n = 20$, $\tau = 10\%$ (e) $t = 0.1$, $n = 75$, $\tau = 10\%$ (f) $t = 0.1$, $n = 75$, $\tau = 5\%$

(g) $t = 0.01$, $n = 20$, $\tau = 10\%$ (h) $t = 0.01$, $n = 75$, $\tau = 10\%$ (i) $t = 0.01$, $n = 75$, $\tau = 5\%$

(j) $t = 0.0$, $n = 20$, $\tau = 0\%$ (k) $t = 0.0$, $n = 75$, $\tau = 0\%$ (l) $t = 0.0$, $n = 75$, $\tau = 0\%$

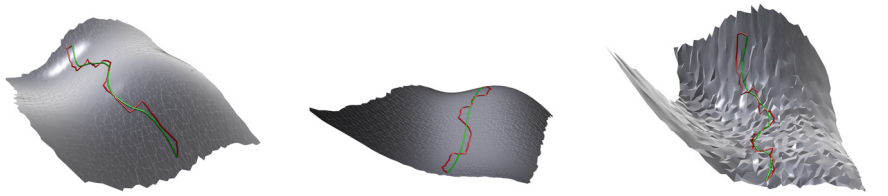
Figure 18: Convergence effect on different tessellations when varying the parameters t, τ, and number of iterations n for a short curve.

Figure 18 represents the comparison to a setting with $\tau = 5\%$: as expected, the result is smoother at the cost of a larger distance from the original curve. However, given a disadvantageous initial curve, the method cannot ensure that the smoothed curve fulfills the condition $t \approx \frac{\kappa_g'}{\kappa_g}$, with κ_g' as the geodesic curvature after the smoothing. This limitation can be easily seen for surfaces that have a hole, an initial curve that wraps around this hole and $t = 0$.



(a) No distortion. (b) With geometric noise. (c) With parametric noise. (d) With both.

Figure 19: Testing robustness toward geometric (distortion in the normal direction) and parametric (distortion in the tangent space) noise for $t = 0.1$ and 20 iterations.



(a) $n = 100$, $t = 0.1$, parametric noise $= 0.01$, $\delta = 0.25$ (b) $n = 100$, $t = 0$, parametric noise $= 0.02$, $\delta = 0.75$ (c) $n = 100$, $t = 0.25$, parametric noise $= 0.02$, geometric noise $= 0.03$, $\delta = 0.75$

Figure 20: Some results of qualitative experiments regarding convergence and robustness on cubic polynomials with different settings. The initial contour is indicated in red, and the smoothed contour is indicated in green.

To investigate the robustness, geometric noise, i.e., displacements in the normal directions, and parametric noise, i.e., displacements in the tangential directions was added. To be comparable, the vertices that coincide with the initial curve were kept at their original position. Therefore, they are not influenced by the noise.

For this experiment, 20 iterations for $t = 0.1$ and $\tau = 10\%$ were used (see Figure 19a). For each scenario, some influence of the noise on the result was noted.

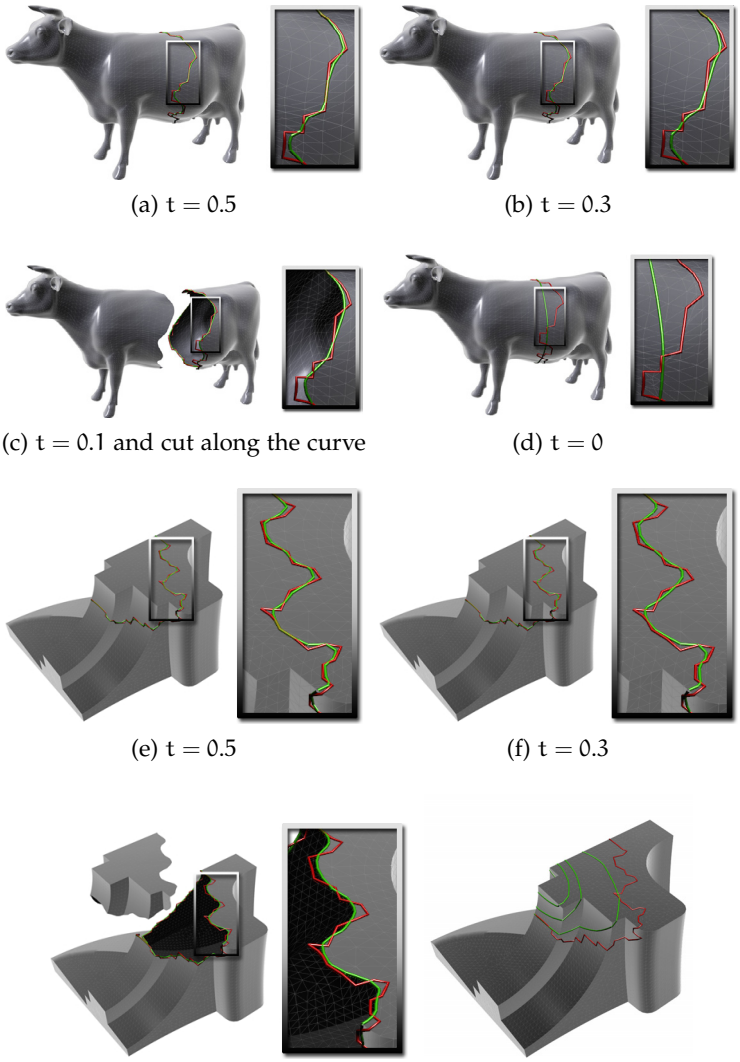
In addition to the qualitative experiment, a quantitative test was performed as well. The experiment, which is strongly inspired by Max [138], was conducted (in the context of normal fitting): random cubic polynomials were generated with coefficients in different ranges. The surfaces are of the form

$$f(x, y) = Ax^2 + Bxy + Cy^2 + Dx^3 + Ex^2y + Fxy^2 + Gy^3.$$

The coefficients A, B, C, D, E, F, and G are all uniformly distributed pseudo-random numbers in the interval $[-\delta, \delta]$ and $x, y \in [-1, 1]$. The domain was subdivided into a 40×40 grid in such a way that the distance of two neighboring vertices with the same x- and y-value is 0.05.

Table 1: Quantitative results of the robustness experiment based on cubic polynomial surfaces. For each parameter setting (#iteration, t, δ, and noise), several quantitative measures between the two resulting curves are compared: κ_g = geodesic curvature before smoothing, κ_g' = geodesic curvature after smoothing, ratio between κ_g and κ_g', which should correspond to t, d = Hausdorff distance, and d % = percentage of deviation of the Hausdorff distance to the straightest geodesic curve.

#Iteration	t	δ	Parametric Noise	Geometric Noise	κ_g	κ_g'	$\frac{\kappa_g'}{\kappa_g}$	$d \cdot 10^2$	d %
20	0.5	0.25	0	0	13.41	6.47	0.48	0.82	58.57
20	0.1	0.25	0	0	13.38	4.98	0.37	1.39	99.28
20	0.0	0.25	0	0	13.40	4.98	0.37	1.40	100
100	0.5	0.25	0	0	13.39	6.48	0.48	0.82	21.10
100	0.1	0.25	0	0	13.41	1.41	0.11	3.17	82.20
100	0.0	0.25	0	0	13.37	0.36	0.03	3.86	100
100	0.5	0.25	0.01	0	21.75	10.37	0.48	0.41	10.63
100	0.1	0.25	0.01	0	20.18	1.98	0.10	3.03	77.22
100	0.0	0.25	0.01	0	22.05	0.45	0.02	3.92	100
100	0.5	0.25	0.00	0.01	14.31	6.68	0.47	0.74	19.39
100	0.1	0.25	0.00	0.01	14.32	1.44	0.10	3.19	83.15
100	0.0	0.25	0.00	0.01	14.17	0.32	0.04	3.83	100
100	0.5	0.75	0.02	0.03	28.52	12.38	0.44	0.57	16.32
100	0.1	0.75	0.02	0.03	30.68	2.98	0.10	1.53	44.03
100	0.0	0.75	0.02	0.03	30.13	0.87	0.03	3.47	100



(a) $t = 0.5$ (b) $t = 0.3$

(c) $t = 0.1$ and cut along the curve (d) $t = 0$

(e) $t = 0.5$ (f) $t = 0.3$

(g) $t = 0.1$ and cut along the curve (h) $t = 0$

Figure 21: The cow and the fandisk dataset with the initial curves (red) and smoothed curves (green) for different values of t. The figures (d) and (h) show the behavior of the curve for $t = 0$. Iterations are performed until a sufficient desired overall curvature value is reached.

Additionally, parametric and geometric noise were added to test for robustness. The noise value γ means that the vertices are translated randomly in a range of $[-\gamma, \gamma]$ in the domain or in the codomain for parametric and geometric noise, respectively. Table 1 shows the results of the experiments with a smooth surface and different types of added noise. The test was performed with several parameters and it presents the averaged total geodesic curvature κ_g before and κ_g' after the smoothing as well as the Hausdorff distance d between the two curves. For every parameter setting, 50 random cubic polynomials were generated. The relative distance d (as a percentage) expresses

the deviation from the assumed maximal Hausdorff distance of the straightest geodesic curve with $t = 0$. Figure 20 shows some results for different parameter settings. According to a quantitative comparison, several observations can be made. For each parameter setting, the geodesic curvature of the curve is decreased while remaining close to its initial curve. Thereby, the number of iterations influences how close the resulting curvature is to the prescribed curvature. A low number results in an increased deviation between κ_g'/κ_g and t compared to a higher number of iterations. Furthermore, it can be seen that the presence of noise leads to a slightly decreased geodesic curvature compared to the non-disturbed surface. The quantitative results, however, demonstrate an overall robustness with respect to noise, which corresponds to observations from the qualitative comparisons.

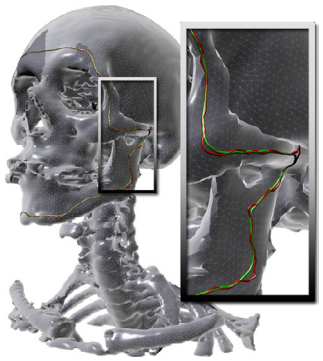
As can be seen, the obtained curves are smooth and robust against noise, and they exhibit reasonable convergence behavior. However, it is obvious that changing the vertex positions by adding geometric noise will not change the geodesic curvature (when keeping the initial curve points at their original position), but this action has, in fact, an influence on the operation space. Because the point can move only along the edges, changing the vertex position will change the intrinsic position of the smoothed curve, i.e., the relative position on the edge could have changed. Moreover, if the vertex positions are distorted in a normal direction, the geodesic curvature will change, and this arrangement leads to a different smoothed curve. Despite the different results, it was observed that the final curve is always smooth. Thus, the algorithm gives robust results even if the underlying surface is distorted in both the tangential and normal directions.

3.2.9.2 *Application to Large Data Sets*

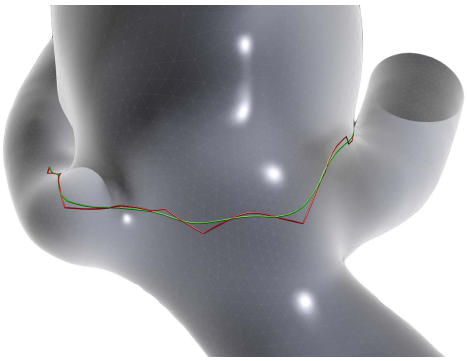
The approach is applied to benchmark surfaces and anatomical surfaces from medical image data sets. The anatomical surfaces exhibit low regularity and a significant amount of noise. In the experiments, only the desired geodesic curvature parameter t were varied; the number of iterations is fixed at 20, and $\tau = 10\%$ is fixed.

Benchmark surfaces. Figure 21 shows results for the *cow* and *fandisk* surface meshes. The initial surface curves are red and the resulting smoothed curves are green. The shapes of the initial curves are nontrivial; their lengths are relatively long and show additional closeups. For the largest $t = 0.5$, a smooth curve was obtained, which is located close to the initial curve. Decreasing t increases the amount of smoothing, and the curves do not move significantly away from the initial curve.

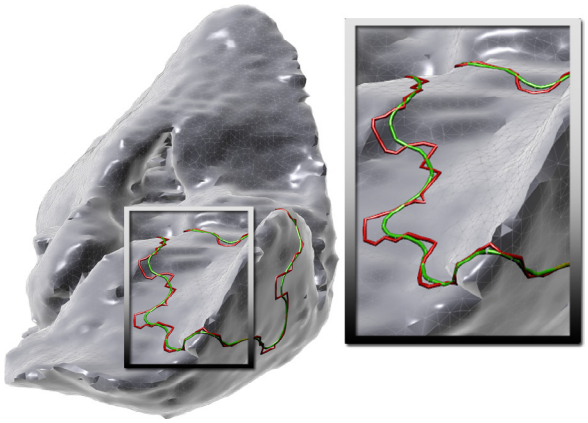
Anatomical surfaces. Figure 22 shows results for the anatomical surfaces: bone structures (Fig. 22a), a cerebral aneurysm (Fig. 22b), and a liver cut (Fig. 22c and Fig. 22d). The smooth curves remain



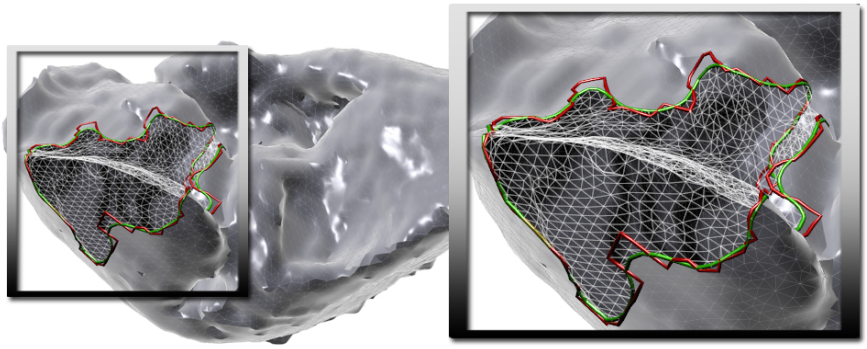
(a) $t = 0.1$



(b) $t = 0.1$



(c) $t = 0.5$



(d) $t = 0.1$

Figure 22: Application to patient-specific medical surface datasets: initial curves (red) and smoothed curves (green) are shown on a complex bone, an aneurysm, and a liver surface dataset, respectively.

close to the initial curves, and no artifacts such as self-intersections were observed. The parameter choice $t = 0.1$ leads to a significant and comprehensible smoothing while closely imitating the original curve.

3.2.9.3 *Comparison to the Spline Approximation*

The algorithm is compared to a global approximation of the initial curve with B-splines. The B-spline approximation in manifolds by Hofer and Pottmann [85, 162] was emulated by assuming and providing a global surface parameterization and resorting to a standard least-squares approximation. Least-squares conformal maps [126] were used to construct the surface mesh parameterization. Cubic B-splines with a uniform knot vector were fitted. The Schoenberg-Whitney conditions are always satisfied by a regularization term, which penalizes the length and (linearized) curvature (see, e.g., [87]). This regularization not only guarantees a solution of the linear systems that arise but also accounts for minimizing exactly the same quantities as in [85]. The initial curve was projected to the parameter space, the B-spline fitting was applied, and the results were mapped back onto the surface. In comparison with the new explicit curve-smoothing algorithm, it obtains similar results (see Figure 66). Note, however, that the B-splines fitting requires either a global parameterization (whose construction is a non-trivial problem on its own) or an adapted iterative optimization scheme with projections to a tangent space in every step [85]. In contrast, the novel method is simpler and leads to similar results for the considered applications. B-spline fitting, however, is more suited for surface curves: for example, the surface curve becomes smoother if (selected) control points are removed [85]. This smoothing in the sense of generating fair curves is not the goal because curves are preferred that remain close to the initial curve. In summary, the minimization of the geodesic curvature is the right choice for the considered applications.

3.2.9.4 *User Feedback*

An informal interview with a domain expert was conducted to gain qualitative user feedback. The domain expert is actively involved in the reconstruction and decomposition of cerebral aneurysm surfaces as well as the exploration of their hemodynamics based on simulated or measured flow data. The surface decomposition involves several geometric operations, such as cutting the aneurysm sac from the parent vessel. The interview was designed to determine if the requirements, defined in Section 3.2.3, were principally met. For several input meshes, the expert should draw an initial curve, which roughly defines the aneurysm neck. Afterwards, the expert was asked to adjust the allowable region. After the smoothing approach was applied to one curve, the aneurysm surface was cut, and its result was evaluated by the domain expert. To obtain a resulting smooth curve, the participant attempted different parameter settings but was mostly satisfied with the values of $t = 0.1$ and $n = 20$. The expert assessed the drawing of the initial curve as being very intuitive and fast. The ad-

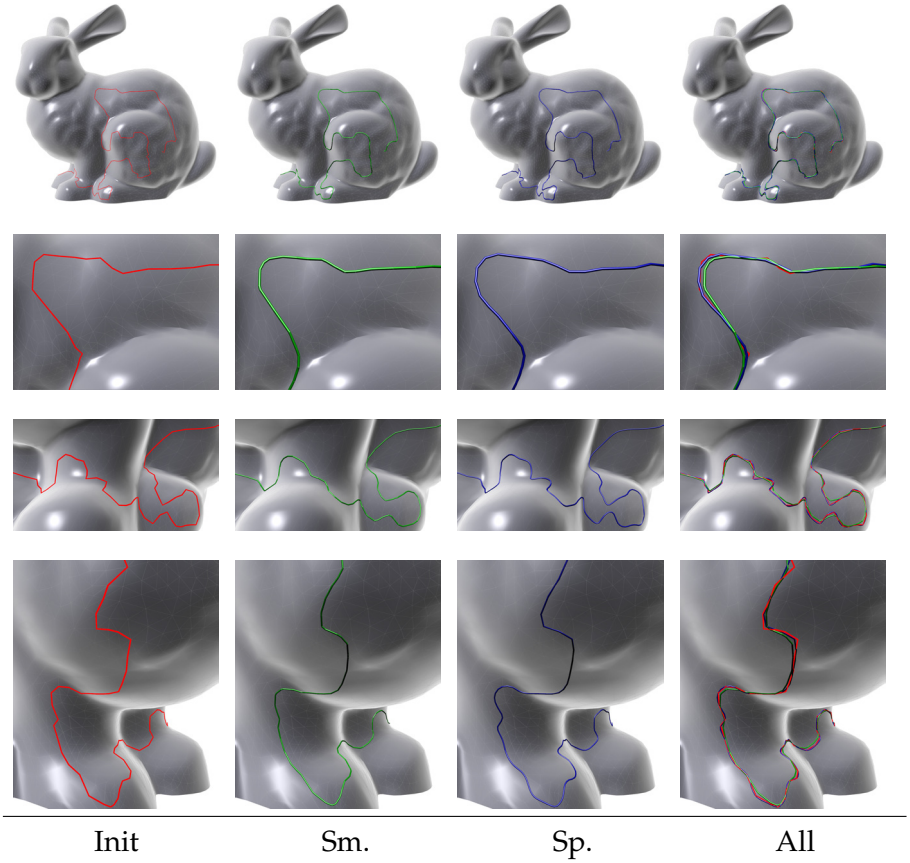


Figure 23: Comparison of the novel curve smoothing approach (Sm.) with the spline approximation method (Sp.) applied on a synthetic surface, demonstrated with three enlarged views. Based on the initial curve (red), the novel approach (green) and the spline approximation (blue) achieve similar results. However, although the spline curve is slightly closer to the initial curve, the novel approach achieves more global smoothness due to the optimization between the geodesic curvature and the closeness to the initial contour.

justment of the allowable region was rated as a pleasant control function to keep the smoothed curve in its eligible region. The smoothing approach was evaluated as being visually pleasant and reasonable as well as time-saving compared to the current definition of the neck contour in the geometric modeling tools. However, the expert suggested providing an overview gallery, which shows different smoothing results based on different parameter settings. This arrangement would lead to an effective selection of appropriate parameter values, such as t and n, depending on the current data set. In summary, the domain expert rated the results positively and gave feedback on improving of the interaction.

3.2.10 *Conclusions*

A novel approach for smoothing surface curves on triangular meshes by reducing the geodesic curvature of the curves was presented. The approach is based on an iterative Laplacian smoothing with a careful construction of the linear operator: It was proved that the curve's curvature decreases during runtime, and this property is used as an abort criterion. Depending on one user-defined parameter, the result gradually changes from a smoothed curve close to the original curve to the straightest geodesic curve. The user can adjust the closeness of the smooth curve to its original shape as well as the deviation from the prescribed geodesic curvature. For these adjustments, default parameters are suggested. The algorithm was tested on both synthetic surfaces and anatomical surfaces from clinical data sets to show robustness in terms of geometric and parametric noise. In this way, the approach is not restricted to triangular meshes but is also applicable to different surface representations. The algorithm fills a gap for the interactive computation of smooth surface curves for cases in which closeness to their initial curve shape is necessary, which was demonstrated for two medical applications. Similar results were achieved in comparison with spline approximation methods, but this approach requires less user effort and does not need a global parameterization. Informal user feedback with a domain expert confirmed the usefulness and robustness of the approach.

Part II

LINE DRAWING TECHNIQUES

Feature Lines

This section is partly based on:

Kai Lawonn and Bernhard Preim
Feature Lines for Illustrating Medical Surface Models:
Mathematical Background and Survey
Visualization in Medicine and Life Sciences (in print), 2014

Kai Lawonn, Rocco Gasteiger and
Bernhard Preim
Qualitative Evaluation of Feature Lines on
Anatomical Surfaces
Bildverarbeitung fuer die Medizin, pp. 187−192, 2013

4 FEATURE LINES

This chapter provides an overview about the most common feature lines. First, general requirements will be discussed. Afterwards, a summary of all feature lines will be presented. Then, all methods will be discussed and compared to each other. Here, medical applications are mentioned and examples are provided where feature lines can be used to enhance visualizations. Next, an evaluation is provided where medical experts assessed the capabilities of certain feature line methods. Finally, this chapter ends with the results and the observations of the evaluation.

4.1 GENERAL REQUIREMENTS OF FEATURE LINES

The generation of feature lines leads to several requirements, which have to be considered for acquiring appropriate results.

Smoothing: Most of the feature line methods use higher order derivatives. Therefore, the methods assume sufficiently smooth input data. For data acquired with laser scanners or industrial measurement processes, methods are applied to generate triangulated surface meshes and smoothness cannot be expected. Discontinuities represent high frequencies in the surface mesh and lead to the generation of distracting (and erroneous) lines. Several algorithms exist, which smooth the surface by keeping relevant features. Depending on the feature line method, different smoothing algorithms can be applied. If the algorithm only uses the surface normals and the view direction, it is sufficient to simply smooth the surface normals. Geometry-based approaches, however, require to smooth the mesh completely. Operating only on scalar values, an algorithm which smoothes the scalar field around a certain region may be applied, too.

Frame-coherence: The application of feature line approaches or in general for non-photorealistic rendering makes it crucial to provide methods that are frame-coherent. This means, during the interaction the user should not be distracted by features that pop out or disappear suddenly. A consistent and continuous depiction of features should be provided in consecutive frames of animation.

Filtering: Feature line algorithms may generate lines on salient regions as well as lines that result from small local irregularities, which may not be necessary to convey the surface shape. Instead, they may

be annoying or distracting. Filtering of feature lines to set apart relevant lines from distracting ones is a crucial part of a feature line generation. User-defined thresholds may control the rate of tolerance for line generation. Some algorithms use an underlying scalar field for thresholding. Lines are only drawn if the corresponding scalar value exceeds the user-defined threshold. Other methods integrate along a feature line, determine the value, and decide to draw the whole line instead of filtering some parts. The filtering technique of each presented feature line generation method will be presented.

4.2 FEATURE LINES

This section presents selected object-based feature line methods. These methods and their limitations will be explained. Further information on line drawings can be found in [164, 170].

4.2.1 *Contours*

A silhouette is referred as a depiction of the outline of an object as this is the original definition by Étienne de Silhouette. The contour is defined as the loci of points where the normal vector and the view vector are mutually perpendicular:

$$\boxed{\langle \mathbf{n}, \mathbf{v} \rangle = 0,}$$

where $\mathbf{n}$ is the normal vector and $\mathbf{v}$ is the view vector which points towards the camera. For the discrete case, the edges were highlighted

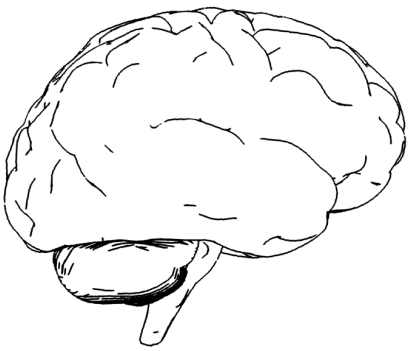


Figure 24: The brain model with contours.

as a contour whenever the sign of the dot product of the view vector with the normals of the incident triangle normals changes. The contour yields a first impression of the surface mesh. On the other hand, it is not sufficient to depict the surface well. The contour is not appropriate to gain a spatial impression of the object. Furthermore, it cannot depict salient regions, for instance strong edges.

Summary: The contour is necessary for gaining a first impression on the shape of the object. Unfortunately, spatial cues, as for instance strong edges, are not depicted.

4.2.2 *Crease Lines*

Crease lines are the set of edges where the normals of incident triangles change strongly. The dihedral angle, i.e., the angle of the normals of the corresponding incident triangles, along the edges is calculated. The edge belongs to a crease line if the dihedral angle exceeds a user-defined threshold τ. As the change of the normals is an indicator of the magnitude of the curvature, one can state that all points contribute to a feature line if the underlying absolute value of the maximum curvature exceeds a threshold:

$$\boxed{\kappa_i \geqslant \tau \text{ or } \langle \mathbf{n}_i, \mathbf{n}_j \rangle \geqslant \tau',}$$

for adjacent triangles with corresponding normals $\mathbf{n}_i, \mathbf{n}_j$. Afterwards,

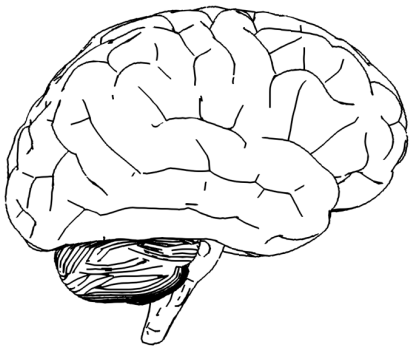


Figure 25: The brain model with crease lines and contours.

all adjacent vertices which fulfill the property are connected. These feature lines need to be computed only once, since they are not view-dependent. Furthermore, these lines are only drawn along edges.

Summary: Crease lines display edges where the dihedral angle is large. Strong edges are appropriately depicted, but if the object has small features, this method is not able to depict only important edges. This is caused by the local determination of the dihedral angle without concerning a neighborhood. Even smoothing the surface mesh would not deliver proper line drawings. Furthermore, this method is only able to detect features on edges.

4.2.3 *Ridges and Valleys*

Ridges and valleys were proposed by Interrante et al. [91] and adapted to triangulated surface meshes by Ohtake et al. [155]. These feature

lines are curvature-based and not view-dependent. The computation is based on the principle curvature κ_1 as well as the associated principle curvature direction $\mathbf{k}_1$ with $|\kappa_1| \geqslant |\kappa_2|$. Formally, ridges and

Figure 26: The brain model with ridges and valleys as well as contours.

valleys are defined as the loci of points at which the principle curvatures assume an extremum in the principle direction:

$$\boxed{D_{\mathbf{k}_1}\kappa_1 = 0.}$$

According to two constraints, the sets of points are called

$$D_{\mathbf{k}_1}D_{\mathbf{k}_1}\kappa_1 \begin{cases} < 0, & \text{and } \kappa_1 > 0\text{: ridges} \\ > 0, & \text{and } \kappa_1 < 0\text{: valleys.} \end{cases} \tag{22}$$

To determine the ridges and valleys, first the principle curvatures and their associated principle curvature directions must be determined, recall Section 2.5.2. Afterwards, the gradient of κ_1 for each vertex is determined, see Section 2.5.3. Finally, the dot product of the gradient and the associated principle curvature direction $\mathbf{k}_1$ is computed. This yields the scalar value of $D_{\mathbf{k}_1}\kappa_1$ for each vertex. Next, ridges and valleys must be distinguished and $D_{\mathbf{k}_1}D_{\mathbf{k}_1}\kappa_1$ for each vertex have to be determined. Here, the gradient of each vertex with the value $D_{\mathbf{k}_1}\kappa_1$ is needed and the dot product of the result with $\mathbf{k}_1$ is determined. Hence, two scalar values per vertex are assessed: $D_{\mathbf{k}_1}\kappa_1$ and $D_{\mathbf{k}_1}D_{\mathbf{k}_1}\kappa_1$. Afterwards, the zero-crossing of the first scalar value is determined, recall Section 2.5.5. The zero-crossings in every triangle are determined for which one condition of Equation 22 holds. The filtering of the lines is again performed by employing a user-defined threshold. The integral along each ridge and valley line is determined according to the underlying curvature. If the magnitude of the integral exceeds the threshold for ridges or valleys, the line is drawn.

Summary: The calculation is solely based on the curvature and therefore view-independent. This method is able to display small features. The filtering depends on the underlying curvature and the length of

the curve. Therefore, a long line with small curvature has also the chance to be drawn as well as a small line with high curvature. This strategy emphasizes also long feature lines. Ridges and valley are very susceptible to noise, since this method is of 3rd order. Therefore, small discontinuities on the surface mesh lead to erroneous derivatives and this error propagates for each further derivative. A crucial task for this method is to guarantee a smoothed mesh to obtain reasonable results. From an artist's point of view, some features may be more highlighted than others from different points of view. This is caused by the different perception of an object and by various light positions. For this task, the ridges and valleys are not appropriate due to the restriction of view-independent results.

4.2.4 *Suggestive Contours*

Suggestive contours are view-dependent feature lines introduced by DeCarlo et al. [46]. They extend the definition of the contour. These

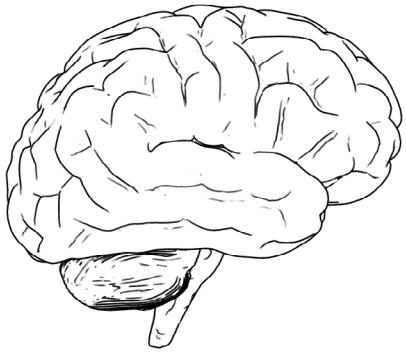


Figure 27: The brain model with suggestive contours and contours.

lines are defined as the set of minima of $\langle \mathbf{n}, \mathbf{v} \rangle$ in the direction of $\mathbf{w}$, where $\mathbf{n}$ is the surface normal, $\mathbf{v}$ is the view vector which points towards the camera, and $\mathbf{w} = (\mathbf{Id} - \mathbf{n}\mathbf{n}^T)\mathbf{v}$ is the projection of the view vector on the tangent plane. Formally:

$$\boxed{D_{\mathbf{w}} \langle \mathbf{n}, \mathbf{v} \rangle = 0 \text{ and } D_{\mathbf{w}} D_{\mathbf{w}} \langle \mathbf{n}, \mathbf{v} \rangle > 0.}$$

Another equivalent definition of the suggestive contours is given by the radial curvature κ_r. It is defined as the curvature in direction of $\mathbf{w}$. As seen in Equation 3, this curvature can be determined by knowing the principle curvature directions as well as the corresponding curvatures. Therefore, the definition of the suggestive contours is equivalent to the set of points at which the radial curvature κ_r is equal 0 and the directional derivative of κ_r in direction $\mathbf{w}$ is positive:

$$\boxed{\kappa_r = 0 \text{ and } D_{\mathbf{w}} \kappa_r > 0.}$$

The filtering strategy is to apply a small threshold to eliminate suggestive contour points where the radial curvature in direction of the

projected view vector is very low. Additionally, a hysteresis threshold is applied to increase granularity.

Summary: Suggestive contours extend the normal definition of the contour. This method depicts zero-crossings of the diffuse light in view direction. This can be seen as inflection points on the surface. This method is of 2nd order only and thus less susceptible to noise. Unfortunately, suggestive contours are not able to depict some sorts of sharp edges, which are in fact noticeable features. For instance, a cube with rounded corners has no suggestive contours.

4.2.5 *Apparent Ridges*

Apparent ridges were proposed by Judd et al. [99]. These feature lines extend the definition of ridges by a view-dependent curvature term. Therefore, a projection operator P is used to map the vertices on a

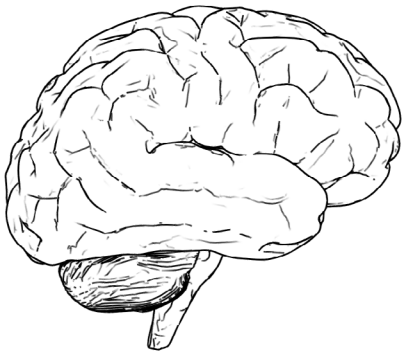


Figure 28: The brain model with apparent ridges.

screen plane V. The orthonormal basis of the screen plane is given by $(\mathbf{v}_1, \mathbf{v}_2)$. Assume having a parametrized surface $f\colon I \subset \mathbb{R}^2 \to \mathbb{R}^3$. Then the projection of f onto V is given by:

$$P(\mathbf{x}) = \begin{pmatrix} \langle \mathbf{v}_1, f(\mathbf{x}) \rangle \\ \langle \mathbf{v}_2, f(\mathbf{x}) \rangle \end{pmatrix}.$$

The Jacobian J_P of P can be expressed as:

$$J_P = \begin{pmatrix} \langle \mathbf{v}_1, \frac{\partial f}{\partial x_1} \rangle & \langle \mathbf{v}_1, \frac{\partial f}{\partial x_2} \rangle \\ \langle \mathbf{v}_2, \frac{\partial f}{\partial x_1} \rangle & \langle \mathbf{v}_2, \frac{\partial f}{\partial x_2} \rangle \end{pmatrix}.$$

In the discrete case with surface meshes, the Jacobian can be expressed by a basis for the tangent plane $(\mathbf{e}_1, \mathbf{e}_2)$:

$$J_P = \begin{pmatrix} \langle \mathbf{v}_1, \mathbf{e}_1 \rangle & \langle \mathbf{v}_1, \mathbf{e}_2 \rangle \\ \langle \mathbf{v}_2, \mathbf{e}_1 \rangle & \langle \mathbf{v}_2, \mathbf{e}_2 \rangle \end{pmatrix}.$$

If a point $\mathbf{p}'$ on the screen plane is not a contour point, there exists a small neighborhood where the inverse of P exists. Normal vectors $\mathbf{n}'$ at a point $\mathbf{p}'$ on the screen plane are defined as $\mathbf{n}'(\mathbf{p}') := \mathbf{n}(P^{-1}(\mathbf{p}'))$. The main idea is to build a view-dependent shape operator S' at a point $\mathbf{p}'$ on the screen as

$$S'(\mathbf{w}') = D_{\mathbf{w}'}\mathbf{n}'$$

where $\mathbf{w}'$ is a vector in the screen plane. The view-dependent shape operator is therefore defined as:

$$S' = S\,J_P^{-1}.$$

Here, the basis of the tangent space expressing S and J_P must be the same. In contrast to the shape operator, the view-dependent shape operator is not a self-adjoint operator, recall Section 2.4.5. Therefore, it is not guaranteed that S' has two eigenvalues, but it has a maximum singular value κ_1':

$$\kappa_1' = \max_{\|\mathbf{w}\|=1} \|S'(\mathbf{w}')\|.$$

This is equivalent to finding the maximum eigenvalue of $S'^{T}S'$ and to take the square root. The corresponding singular eigenvector $\mathbf{t}'$ is called the maximum view-dependent principle direction. The rest of the method is similar to the ridge and valley methods. Formally, apparent ridges are defined as the loci of points at which the view-dependent principle curvature assumes an extremum in the view-dependent principle direction:

$$\boxed{D_{\mathbf{t}'}\kappa_1' = 0 \;\text{ and }\; D_{\mathbf{t}'}D_{\mathbf{t}'}\kappa_1' < 0.}$$

The sign of κ' is always positive. To distinguish between ridge lines and valley lines, the sign of the object-space curvature has to be compared:

$$\kappa_1 \begin{cases} < 0, & \text{ridges} \\ > 0, & \text{valleys.} \end{cases}$$

The calculation of the directional derivative is different from the other methods. This calculation is performed with finite differences. Therefore, the singular eigenvector $\mathbf{t}'$ are transformed to object space $\mathbf{t}$ using the corresponding basis of the associated vertex i. Furthermore, the opposite edges of the vertex are needed and two points $\mathbf{w}_1$, $\mathbf{w}_2$ on the edges are determined such that $\mathbf{t}$ and the edges are orthogonal and $\mathbf{w}_1$, $\mathbf{w}_2$ are the dropped perpendiculars of $\mathbf{t}$ to the corresponding edges. The directional derivatives are determined by averaging the finite differences of the curvatures between $\mathbf{p}_i$ and $\mathbf{w}_1$, $\mathbf{w}_2$. The curvature of $\mathbf{w}_1$ and $\mathbf{w}_2$ is assessed by linear interpolation of the endpoints

of the associated edge. Having the principle view-dependent curvature direction $\mathbf{t}'$, it needs to be consistent over the mesh because it is not well-defined. Therefore, $\mathbf{t}'$ is flipped in opposite direction whenever it does not point to the direction where the view-dependent curvature is increasing. The zero-crossings are determined by checking if the principle view-dependent curvature directions of the vertices along an edge point are in the same direction. Only in this case there exists no zero-crossing. Pointing in different directions means that the enclosing angle is greater than 90 degree. The zero-crossing is determined by interpolating the values of the derivatives. To locate only maxima, a perpendicular is dropped from each vertex to the zero-crossing line. If the perpendiculars of the vertices of an edge make an acute angle with their principle view-dependent curvature directions, the zero-crossing is a maximum. Otherwise, the zero-crossing is a minimum. To eliminate unimportant lines, a threshold based on the view-dependent curvature is used.

Summary: Apparent ridges incorporate the advantages of the ridges and valley lines as well as the view dependency. They extend the ridge and valley definition by introducing view-dependent curvatures. This method is able to depict salient regions as sharp edges. Unfortunately, the 3rd order computation leads to low frame rates and to visual clutter if the surface mesh is not sufficiently smoothed.

4.2.6 *Photic Extremum Lines*

Photic extremum lines (PELs) were introduced by Xi et al. [209]. These feature lines depict regions of the surface mesh with significant variations of illuminations. This method is based on the magnitude of the

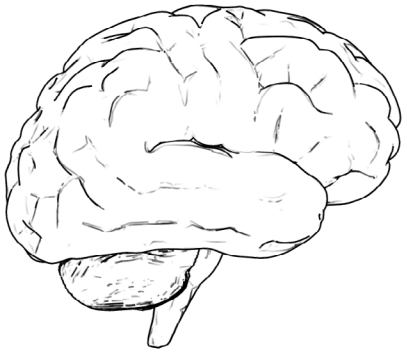


Figure 29: The brain model with photic extremum lines.

light gradient. Formally, these lines are defined as the set of points where the variation of illumination along their gradient direction reaches a local maximum:

$$\boxed{D_{\mathbf{w}}\|\nabla f\| = 0 \;\text{ and }\; D_{\mathbf{w}}D_{\mathbf{w}}\|\nabla f\| < 0,}$$

with $\mathbf{w} = \frac{\nabla f}{\|\nabla f\|}$. Normally, f is used as the headlight illumination: $f \coloneqq \langle \mathbf{n}, \mathbf{v} \rangle$ with $\mathbf{n}$ as the normal vector and $\mathbf{v}$ as the view-vector. PELs have more degrees of freedom to influence the result by adding more light sources. Thus, the scalar value of f changes by adding the light values of the vertices by other lights. Noisy photic extremum lines are filtered by a threshold which is based on the integral of single connected lines. The strength T of a line with points $\mathbf{x}_0, \ldots, \mathbf{x}_n$ is determined by:

$$T = \int \|\nabla f\| = \sum_{i=0}^{n-1} \frac{\|\nabla f(\mathbf{x}_i)\| + \|\nabla f(\mathbf{x}_{i+1})\|}{2} \|\mathbf{x}_i - \mathbf{x}_{i+1}\|.$$

If T is less than a user-defined threshold, the line is canceled out.

Summary: Photic extremum lines are strongly inspired by edge detection in image processing and by human perception of a change in luminance. It uses the variation of illumination. The result may be improved by adding lights. Besides the filtering strategy to integrate over the lines and accumulate the magnitude of the gradient, the noise can also be reduced by adding a spotlight that directs to certain regions. Nevertheless, smoothing is necessary to gain reasonable results. Here, the smoothing of the normal is sufficient as the computation is mainly based on the normals. However, the computation has high performance costs. The original work was improved by Zhang et al. [212] to significantly reduce the runtime.

4.2.7 *Demarcating Curves*

Demarcating curves were proposed by Kolomenkin et al. [111]. These feature lines are defined as the transition of a ridge to a valley line. To determine these lines, the derivative of the shape operator has to

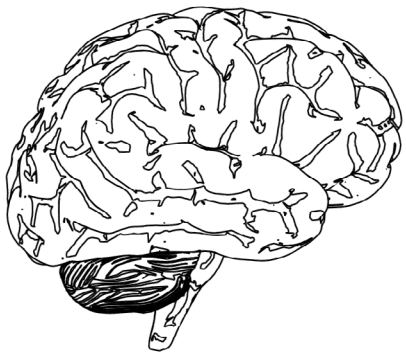


Figure 30: The brain model with demarcating curves and contours.

be calculated, recall Equation 12:

$$C = \begin{pmatrix} D_{\mathbf{v}}S & D_{\mathbf{w}}S \end{pmatrix}.$$

The demarcating curves are defined as the set of points where the curvature derivative is maximal:

$$\boxed{\langle \mathbf{w}, S\mathbf{w} \rangle = 0 \quad \text{with} \quad \mathbf{w} = \arg \max_{\|\mathbf{v}\|=1} D_{\mathbf{v}}\kappa.}$$

The values for $\mathbf{w}$ can be analytically found as the roots of a third order polynomial. This is obtained by setting $\mathbf{v} = \binom{\sin(\theta)}{\cos(\theta)}$ and combining this with Equation 13. A user-defined threshold eliminates demarcating curves, if they exceed the value of $D_{\mathbf{w}}\kappa$.

Summary: Demarcating curves are view-independent feature lines displaying regions where the change of the curvature is maximal. Therefore, higher-order derivatives are used. A $2 \times 2 \times 2$ rank-3 tensor is determined. This method can be used to illustrate bumps by surrounding curves. The advantage of the method is to enhance small features. Especially when combined with shading, the authors claimed that their approach has its strength in illustrating archaeology objects where specific details are important, e.g., old scripts. For this application, view-dependent illustration techniques are not recommended because details need to be displayed for every camera position. Contrary, due to higher-order derivatives, the method is sensitive to noise and is not well suited for illustrative visualization.

4.2.8 *Laplacian Lines*

Laplacian lines were proposed by Zhang et al. [213]. The introduction of these lines was inspired by the Laplacian-of-Gaussian (LoG) edge detector in image processing and aims at a similar effect for surface meshes. The idea of the LoG method is to determine the Laplacian of

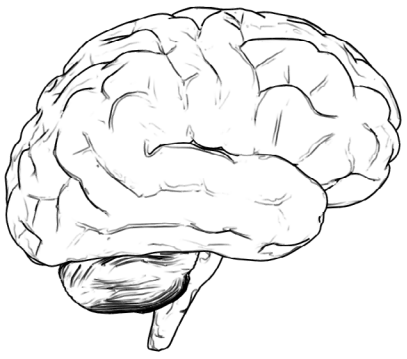


Figure 31: The brain model with Laplacian lines.

the Gaussian function and to use this kernel as a convolution kernel

for the image. Laplacian lines calculate the Laplacian of an illumination function f and determine the zero-crossings as feature lines. To remove noisy lines, the lines are only drawn if the magnitude of the illumination gradient exceeds a user-defined threshold τ:

$$\boxed{\Delta f = 0 \text{ and } \|\nabla f\| \geqslant \tau,}$$

where Δ is the discrete Laplace-Beltrami operator on the surface mesh and f is the illumination with $f := \langle \mathbf{n}, \mathbf{v} \rangle$. Here, the discrete Laplace-Beltrami operator with the Belkin weights is used, as introduced in Section 2.5.4. The advantage of this method is the simplified representation of the Laplacian of the illumination:

$$\begin{aligned} \Delta f(\mathbf{p}) &= \Delta \langle \mathbf{n}, \mathbf{v} \rangle \\ &= \langle \Delta \mathbf{n}, \mathbf{v} \rangle. \end{aligned}$$

Here, $\Delta \mathbf{n}$ is the vector Laplace operator in the Euclidean space. This is just a composite of the Laplacian of the different components. Thus, the algorithm consists of a preprocessing step to calculate the Laplace-Beltrami operator with the Belkin weights of the components of the normal $\Delta \mathbf{n}$. During runtime, the algorithm detects the zero-crossings of $\langle \Delta \mathbf{n}, \mathbf{v} \rangle$ and checks if the magnitude of $\|\nabla f\|$ exceeds the user-defined threshold.

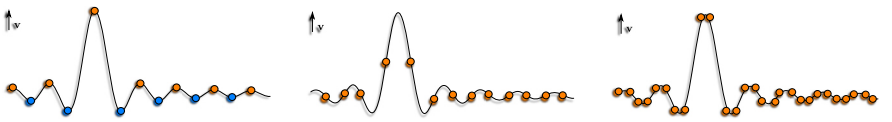
Summary: Laplacian lines are strongly inspired by edge detection algorithms in image processing. This method is based on the Laplacian-of-Gaussian. Basically, the method searches for zero-crossings in the Laplacian of the illumination. The computational effort can be simplified by a preprocessing step. Thus, interactive frame rates for geometric models of moderate size are possible during the interaction. Similar to other higher order methods, this approach also assumes well smoothed surface normals. The Belkin weights for the Laplace-Beltrami operator have a smoothing effect for the Laplacian line generation. This method illustrates sharp edges well, but is not suitable for round corners.

4.3 DISCUSSION AND COMPARISON

This section deals with general properties of the different feature line methods. The different approaches are discussed to derive first recommendations which method may be used for which kind of geometry. First, all feature line methods, different properties, and the order of the corresponding method are listed in Table 2. Furthermore, in Figure 32, the higher-order feature lines are illustrated on an analytic function.

Table 2: List of different feature line methods with derivative order and view dependency.

Name	Order	View-dep.
Contours	1	yes
Crease Lines	1	no
Ridges & Valleys	3	no
Suggestive Contours	2	yes
Apparent Ridges	3	yes
Photic Extremum Lines	3	yes
Demarcating Curves	3	no
Laplacian Lines	3	yes



(a) Ridges and Valleys, Apparent Ridges

(b) Suggestive Contours, Demarcating Curves

(c) Photic Extremum Lines, Laplacian Lines

Figure 32: Drawing of an analytic function with illustrated feature line positions. In (a) the ridges are denoted in orange and the valleys are illustrated in cyan. For this function with fixed view direction, the apparent ridges coincide with ridges and valleys. In (b) the suggestive contours and the demarcating curves are the same, colored in orange. In (c) the photic extremum lines and the Laplacian lines, also colored in orange, coincide.

The benefit of feature lines is motivated by the visual perception. In [135] it is stated that the first stage of the assessment of the shape is done by extracting features, such as contours. These characteristics help to understand the shape. The illustration of shapes with feature lines cannot be seen as an alternative to shading. It is rather an additional concept. Kolomenkin et al. [111] showed that their demarcating lines support the shading and can extract text from archaeology objects. However, for examining structures where the whole object inherits important information, feature lines should not be used solely. For data where the scene can be divided into focus and context objects, feature lines can be applied to the context objects. Furthermore, feature lines can also be used to enhance focus with additional shading.

Depending on the underlying model, different techniques are recommended. Most of the feature lines are able to depict the contour, but this depends strongly on the bending of the surface at the contour. Especially apparent ridges and photic extremum lines are able to

Table 3: List of supported feature by the methods. The different features are illustrated in Figure 33.

Name	Sharp Edges	Round Edges	Bumps (s.w.)	Bumps (top)	Contour	Deformation
Contour	☒	☒	☒	☒	☑	☑
Crease Lines	☑	☒	☒	☒	☒	☑
Ridges & Valleys	☑	☑	☒	☑	☒	☒
Suggestive Contours	☒	☒	☑	☑	☒	☑
Apparent Ridges	☑	☑	☑	☑	☑	☒
Photic Extremum Lines	☑	☒	☒	☑	☑	☑
Demarcating Curves	☒	☒	☒	☑	☒	☒
Laplacian Lines	☑	☒	☒	☑	☑	☒

draw contour lines, but in the experiments activating the contour enhances the visual impression because some parts of the contour were missing. If the surface model is an assembly with sharp edges, ridges

and valleys or apparent ridges are recommended. These features often appear in medical models like implants or prostheses. For simple models with only a few sharp edges, crease lines may be appropriate as well. If the models have a lot of round edges the selection of a feature line method is a matter of taste. These features appear in models like vascular surfaces or organs. For scenarios where it is important to illustrate details for every camera position, ridges and valleys as well as demarcating curves are recommended. From an artistic point of view, suggestive contours, photic extremum lines, and Laplacian lines should be chosen. The reason for this recommendation is that especially for a rounded cube the photic extremum lines and Laplacian lines generate double lines around the feature to denote the rounded edge. If the user wants to visualize the line along the edge, the ridges and valleys or the apparent ridges should be used. For this case, the crease line approach is not useful because it depicts only edges with specific greater value of the dihedral angle. Therefore, too many lines may be generated. If the surface has many crevices, the suggestive contours should be used. They illustrate the inflection points of valleys. Round corners are often represented in many organic structures like livers or bones, see Figure 34 for a femur model or a skull model.

Table 3 lists all possible features and Figure 33 shows the different features. Please note that the assessment of the suitability of a method – marked in the table – necessarily is a subjective assessment by the author and two artists. For instance, regarding the property whether the methods are able to detect round edges, it is meant if it displays the specific round feature. As already mentioned, it does not reflect the ability to enhance the round edge from an illustrative or artist point of view. This concerns the ability to depict bumps. In agreement with artists, the bump shown from a sideway (s.w.) perspective would be illustrated such that it depicts the smooth transition from the ground to the dent. The drawing of the surrounding circle of the bump is not desirable as it conveys a sharp transition from the bump to the ground. For bumps shown from the top perspective it is sufficient if a round circle is drawn. Bumps can occur as polyps or blebs on a cerebral aneurysm. Especially blebs are important anatomical features to be detected because they are an indicator for rupture. Blebs can also occur as polyps in CT colonography.

Also the property *deformation* is listed in the table. This characteristic means if the corresponding method is able to illustrate the features of deformable surfaces, e.g., animated objects, in real-time. As an example, Oeltze et al. [154] analyzed myocardial perfusion data. The focus lies on the examination of the infarction scar on the left ventricle. In this paper, the left ventricle is illustrated as context information. Using the time-dependent data, it would also be possible to illustrate the context information with some feature line methods during the animation.

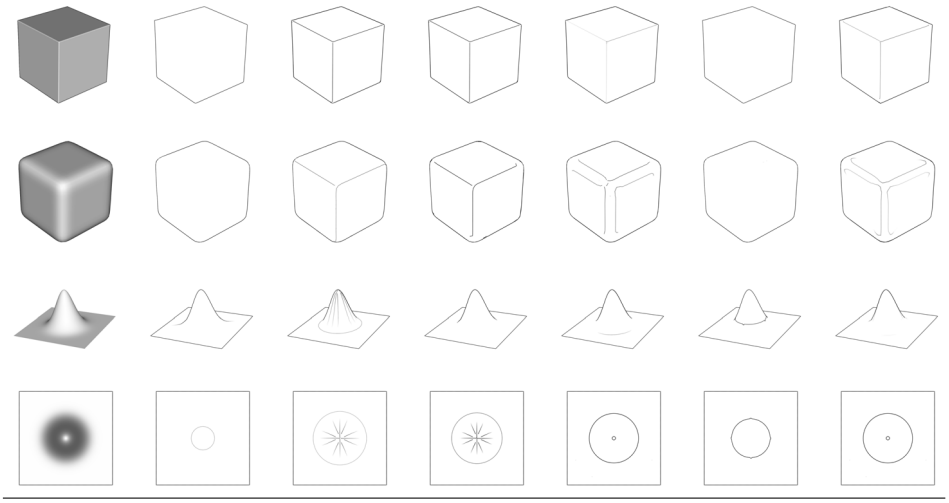


Figure 33: Different surface features are illustrated with shading (SH) and with different higher-order feature line methods: suggestive contours (SC), ridge and valley (RV), apparent ridges (AR), photic extremum lines (PEL), demarcating curves (DC), and Laplacian lines (LL).

For example, suggestive contours have two definitions of how to assess the feature lines. One is curvature-based and the other is light-based. With the second definition, no preprocessing is needed to assess the curvature and the principle curvature directions. This is in contrast to ridges and valleys and apparent ridges. Therefore, these algorithms are not able to compute the feature lines during the deformation. Photic extremum lines are also able to compute the feature lines during runtime because of the light and view dependency. The Laplacian lines need to precompute the Laplacian of the normals. Hence, this method is not suited for deformations.

Furthermore, Figure 34 shows some exemplary models illustrated with higher-order feature lines. Three typical models in the discrete differential geometry field (cow, Buddha, Max Planck) as well as three medical image data models (brain, femur, skull) are presented.

In summary, current feature lines are not suitable for the depiction of anatomical structures directly derived from medical image data because the underlying surfaces are too noisy. Advanced smoothing algorithms are necessary to reduce artifacts, but preserve important anatomical structures. For the depiction of a sparse representation of the model in a context-aware manner, the feature line methods can be used.

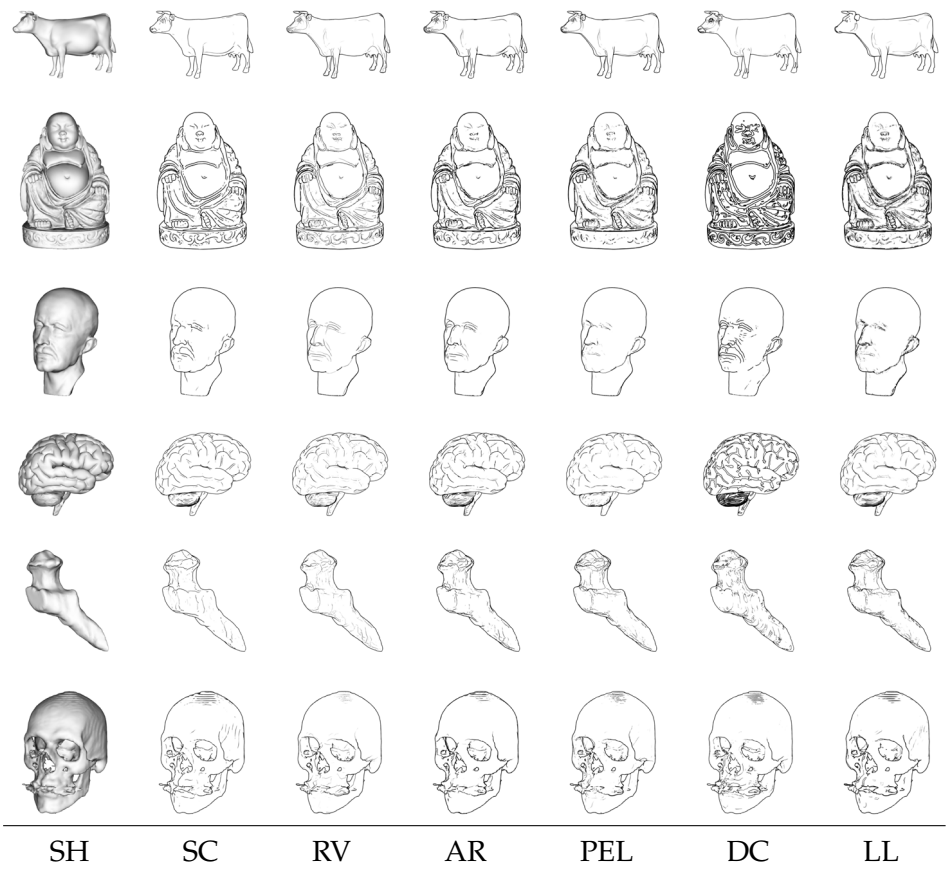


Figure 34: Selected models depicted in shading and higher-order feature lines.

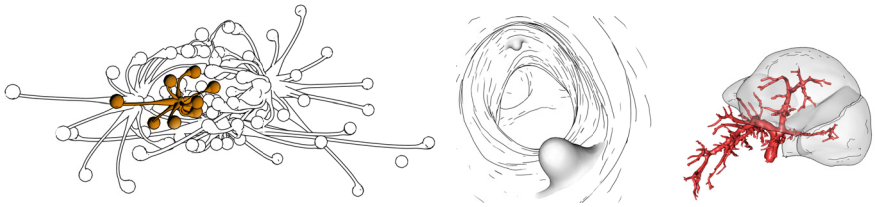


Figure 35: Different medical application fields where feature lines can be used to illustrate surrounding objects.

4.3.1 *Medical Application*

Feature lines are illustrative visualization techniques that can enhance shading or that provide an alternative in the focus-and-context visualization. Glaßer et al. [216] presented an approach to visualize 3D cluster results. Here, the medical researcher can analyze the whole 3D scene with different cluster results where he can also select interesting objects. The surrounding objects become context information. Thus, it is proposed to illustrate them with feature lines. In this case,

the contour can be used only because the objects do not inherit many features. Figure 35a illustrates the main object with context objects, i.e., unselected objects illustrated with feature lines.

In the field of endoscopic views, the identification of polyps is often necessary. Once the polyps are detected, they can be illustrated in such a way that the endoscopic views are used for context information. In Figure 35b, suggestive contours are recommended for the vessel and diffuse shading for the polyps.

In Figure 35c, the portal vein and three liver segments are visualized. The portal vein is illustrated in diffuse shading in red. The liver segments are visualized in diffuse shading with transparency and photic extremum lines.

4.4 EVALUATION

A qualitative evaluation was performed to assess the feature line capabilities in capturing important surface features compared with surface shading. The evaluation was conducted with two physicians and one medical researcher who are familiar with medical visualizations. Three representative surface models are chosen:

- a cerebral aneurysm,
- a trachea seen from an endoscopic view, and
- a liver.

All surface models are derived from clinical image data and binary segmentation masks. Thus, they exhibit surface noise and other artifacts like staircases. Since all four feature line methods are based on higher order derivatives, they are sensitive to noise and the underlying tessellation. To ensure a reliable comparison between the different methods and to reduce these artifacts, each surface model is smoothed with a low pass filter according to the recommendations in Bade et al. [5]. Furthermore, an equal and appropriate degree of tessellation among the surface models is ensured. As different medical models were chosen for a comparison, the evaluation was not conducted with creases lines, demarcating curves, and Laplacian lines. Comparing suggestive contours with demarcating curves and photic extremum lines with Laplacian lines in Figure 32 yields similarities. Furthermore, the Laplacian lines need a preprocessing step with user-defined values and therefore, they are not appropriate for the clinical routine. Demarcating curves and creases lines are susceptible to noise and therefore, they are also not appropriate for medical data sets.

The evaluation itself was conducted in two parts. In the first part, each participant was shown the shaded surface models, which could be explored interactively to gain an impression of important surface features. For each surface model one out of four feature line methods

(ridges and valleys (RV), suggestive contours (SC), apparent ridges (AR), photic extremum lines PEL) was overlayed successively. For each method, the participants were asked to adjust the corresponding threshold until the resulting feature lines capture as much as possible surface features compared to the shaded representation. Thereby, a tradeoff between inherent feature lines and *false-positive* feature lines resulting from surface artifacts should be considered. During the evaluation, the participants were also able to hide the shaded surface model. At the end of each adjustment, the final threshold was recorded. The second part consists of a visual comparison and a qualitative assessment between the feature line methods. Based on the recorded threshold, the participants should assess which method is more appropriate to capture surface features and which limitations they observed.

4.5 RESULT

In Figure 40 the three shaded surface models are shown. For each model the underlying rows represent the feature line representation with one of the four techniques. Thereby, the result of the best-choice for the given technique is rated by the participants. For the aneurysm model, participant #1 (P#1) observed that the generated lines by RV and SC are not sufficient to gain a 3D impression. For AR and PEL, the resulting lines are reasonable, but some lines are distracting. Finally, P#1 preferred the PEL method. P#2 and P#3 rated the result of the SC method as their favored technique. For RV, AR, and PEL P#2 stated that most of the generated lines are not meaningful or distracting, but they depict parts of the bifurcation well. Additionally, P#3 mentioned that AR produces lines on small vessel parts which lead to the impression that the vessel is very wrinkly.

The inner view of the trachea has two main features: the elongated structure and the bifurcation. P#1 stated that RV gives no satisfactory impression of the 3D structure. Apart from that, SC, AR, and PEL depict the elongated structures but fail to enhance the bifurcation. Although PEL produces more unnecessary lines which are distracting, P#1 preferred the result of the PEL method. P#2 and P#3 preferred the SC method because it conveys the elongated structure as well as the tracheal cartilage. Furthermore, the resulting lines depict also the bifurcation appropriately. Both participants noted that they could not figure out the bifurcation when using the RV, AR, or PEL method.

The liver model failed for the illustration since too many distracting lines are generated. This is probably due to the fact that the liver shape has few prominent surface features and the lines emphasize artifacts from image acquisition and surface generation instead of real anatomical features. Thus, without the shaded underlying model the participants were not able to recognize the model as a liver. From an

illustrative point of view, P#1 chose the RV method as his favored. P#2 did not favor a particular technique and noted that it is necessary to rotate the model in order to gain an impression of the model. Finally, P#3 chose SC as the favored method but noticed that the differences between the feature line methods are not significant when using the liver model.

4.6 DISCUSSION

The results of the evaluation can be summarized in two conclusions. First, reasonable depictions of patient-specific surface models with

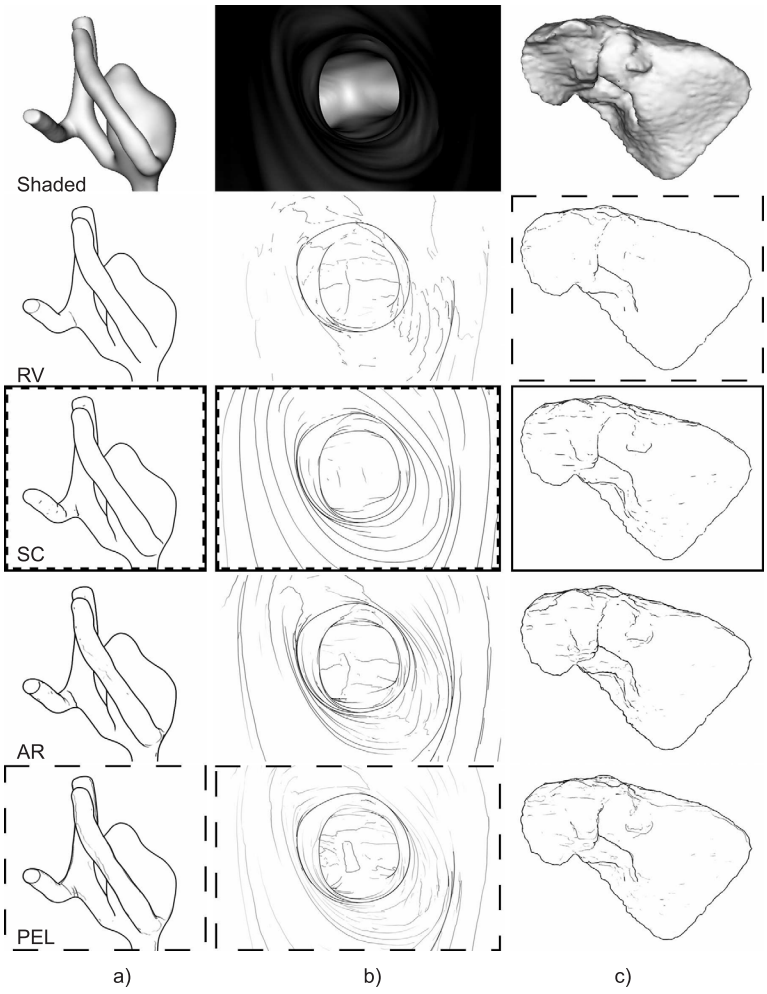


Figure 36: Application of the four feature line techniques applied on: (a) a cerebral aneurysm, (b) an inner view of a trachea, and (c) a liver surface. The resulting images are obtained by best-choice adjustments of the domain experts. Favored results of the experts are depicted with corresponding borders. The dashed line border stands for the best-choice by P#1, e.g. (a) and PEL. The dashed points represent the choice by P#2, e.g. (a) and SC. The line border depicts the favored result by P#3, e.g. (b) and SC.

current feature line methods are obtained only if the models exhibit a smooth and regularly tessellated surface. Due to the higher-order derivations of the methods they are sensitive to surface noise and artifacts. Advanced smoothing and remeshing algorithms are necessary to reduce these artifacts but preserve important anatomical surface shape and features. Thereby, the user has to find a tradeoff between surface shape and plausible resulting feature lines. However, for some cases it seems that current feature line methods are not able to display important features of the underlying model. Additional surface shading is needed. Furthermore, the evaluation shows a subjective rating in terms of choosing a preferred method. It seems that the SC method tends to be the most expressive technique. The second conclusion considers the application of feature line visualizations. Since they are able to provide a sparse representation of the underlying model, they can be used for context-aware medical illustrations in which the model should not be in the focus, but serves as anatomical context.

4.7 CONCLUSION

The most common feature line methods for object space-based presentations of 3D meshes were summarized as they are frequently used in medicine and molecular biology. All the calculation and the mathematical background were provided. Additionally, a qualitative evaluation with medical experts was performed to assess the capabilities of capturing important surface information. In summary, the evaluation put two results on record. First, feature lines cannot serve as an alternative to shading. Feature lines do only capture important regions whereas they cannot convey a spatial impression on the surface. Second, it seems that the suggestive contours are the most expressive technique. Additional surface shading seems necessary to provide all surface information. These results serve as a starting point for improvements on line drawing techniques as well as a new kind of shading.

Streamlines for Illustrative Visualization

This section is partly based on:

Kai Lawonn, Tobias Moench and
Bernhard Preim
Streamlines for Illustrative Real-time Rendering
Computer Graphics Forum, pp. 321-330, 2013

Kai Lawonn, Patrick Saalfeld and
Bernhard Preim
Illustrative Visualization of Endoscopic Views
Bildverarbeitung fuer die Medizin, pp. 276-281, 2014

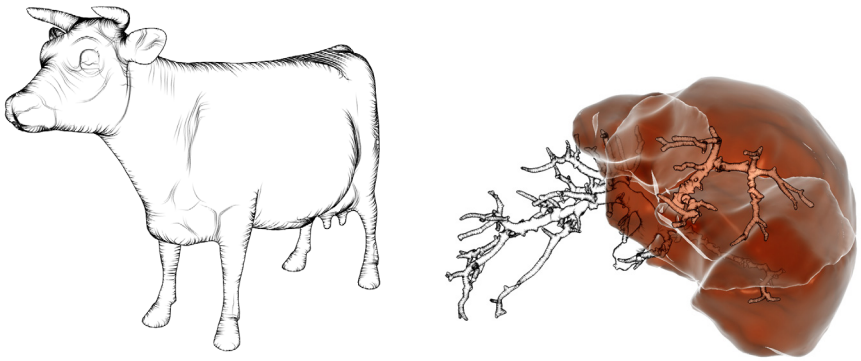
5

STREAMLINES FOR ILLUSTRATIVE VISUALIZATION

5.1 INTRODUCTION

As a result of the previous chapter, feature lines were not able to depict all important surface information, e.g., the spatiality. In contrast, hatching techniques convey a spatial impression on the surface, but may not emphasize important regions. This chapter introduces *ConFIS* – a new illustrative visualization technique to overcome the disadvantages of hatching and feature lines. ConFIS stands for the keywords: **Con**tours, **F**eatures, **I**llustration, **S**treamlines (see Fig. 37). This method is motivated by the peculiarities in medicine, but it is not restricted to that domain. It will be shown that common feature line techniques cannot convey the specific shape of several patient-specific anatomical surfaces, e.g., endoscopic views. On the one hand, hatching techniques allow for a better spatial perception in such endoscopic views. On the other hand, they usually draw hatching patterns allover the surface and miss to depict salient regions well. ConFIS is designed to remedy these issues. The approach was qualitatively evaluated with two physicians and three medical researchers on various models. The goal of the evaluation was to assess, with which illustrative visualization technique the domain experts can better infer the surface shape. They had to compare the ConFIS-based illustrations to surface shading and the most commonly used feature line and hatching methods. Afterwards, the evaluation showed that ConFIS is more appropriate on endoscopic views than feature lines or hatching methods. This evaluation was extended with more test persons and more alternative techniques. Here, another hatching method was used to compare ConFIS on endoscopic views only. This chapter can be summarized with the following contributions:

- A novel view-dependent illustrative visualization method. Explicit streamlines on the triangular surface mesh are employed and drawn in real-time.
- Feature regions are determined by resolving the minima and maxima of the mean curvature scalar field.
- Expressive illustrations on endoscopic anatomical surfaces.
- A qualitative evaluation with medical experts has been conducted to compare ConFIS with existing illustration techniques.



(a) Cow (b) Three Liver Segments & Portal Vein

Figure 37: ConFIS is visually pleasing and conveys the surface model and the volume reasonable.

5.2 METHOD

ConFIS is based on streamlines to illustrate the surface model. For this reason, the regions, where streamlines should be seeded, will be described first. Second, the streamline calculation will be explained. Thus, this section will be divided in three parts:

1. **Contour margin:** The term *contour margin* will be explained.
2. **Feature regions:** A curvature-based definition for a feature region is described.
3. **Streamlines:** The determination of explicit streamlines will be detailed.

5.2.1 *Contour Margin*

The common edge of two adjacent triangles is highlighted if the signs of the dot products between the oriented face normals and the view vector change. Besides the contour line, also streamlines are drawn at contour triangles. To provide frame-coherent interaction, streamlines are seeded at triangles which lie inside a *contour margin*. The *contour margin* is defined by the curvature-based method of Kindlmann et al. [108]. Triangles are included in the *contour margin* if the dot product of the normal and the view vector is less than $\sqrt{T\kappa_v(2-T\kappa_v)}$. Here, κ_v denotes the normal curvature in direction of the view vector and

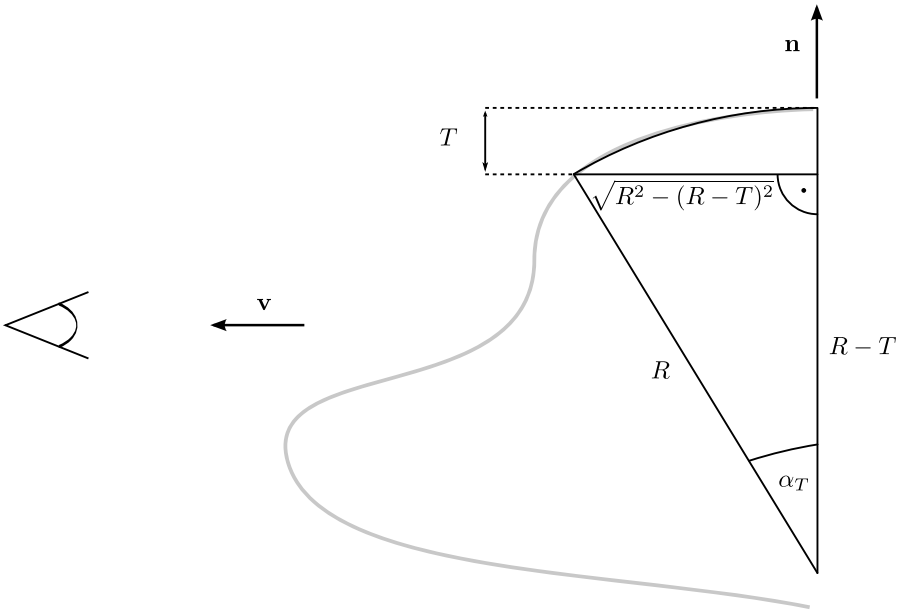


Figure 38: Condition for the dot product of the normal and the view vector depending on the thickness T.

T denotes the thickness of the contour margin. The determination is based on the view vector and the normal, see Figure 38:

$$\begin{aligned}
|\langle \mathbf{v}, \mathbf{n} \rangle| &\leqslant \sin(\alpha_T) \\
&\leqslant \frac{\sqrt{R^2-(R-T)^2}}{R} \\
&\leqslant \sqrt{\frac{2RT-T^2}{R^2}} \\
&\leqslant \sqrt{\frac{T}{R}\cdot\left(2-\frac{T}{R}\right)} \\
&\leqslant \sqrt{T\kappa_v(2-T\kappa_v)}.
\end{aligned}$$

This method provides a homogeneous *contour margin* in image space. Additionally, the opacity of the streamlines changes depending on the length to provide a convenient fade-off during the interaction. For an extensive overview about contours, the reader is referred to Isenberg et al. [93].

5.2.2 *Feature Regions*

Definition 5.2.1 (Maxima and minima on a scalar field)**.** *Two cases will be considered for the maxima and minima on a scalar field.*

- *Continuous Case: Maxima and minima can occur where the gradient of the scalar field vanishes.*
- *Discrete Case: The gradient of the scalar field for each vertex of a triangle is determined. The three dot products between the gradients*

are calculated. A maximum or a minimum occurs if two dot products are negative.

A feature streamline is seeded at a triangle t if the following properties for the mean curvature field (MCF) are fulfilled:

i. The MCF has a maximum or a minimum at t.

ii. The MCF at t exceeds a user-specified threshold.

5.2.3 *Streamlines*

Streamlines are seeded at every contour triangle and within a defined margin around the contour. Thus, the following tasks have to be solved:

- Selection of the vector field for streamline calculation.
- Select an appropriate streamline starting point.
- Estimate the maximum streamline length.
- Calculate the streamline.
- Use an adaptive step size for the streamline generation.

The vector field for streamline calculation: A suitable curvature-based vector field for streamline calculation is chosen. In order to compute curvatures on a discrete triangle mesh, the algorithm from Rusinkiewicz [169] is used, since it yields accurate and robust results even on irregularly tessellated surfaces, recall Section 2.5.2. At umbilic vertices, the principle curvature directions (PCDs) are set to the zero vector to exclude the streamlines from seeding. Four vectors are assigned at the vertices and the triangle – namely two PCDs, each with two possible signs. For each triangle, only e_1 is used where e_1 is the PCD which corresponds to the maximum principle curvature (PC). The e_1 of each triangle is compared with the four vectors of their adjacent vertices. The dot product between e_1 and the four vectors of the first vertex is determined. The vector which corresponds to the largest dot product is selected, see Figure 39. The resulting vector belongs to the final vector field to ensure that is smooth. This procedure is repeated with the other two vertices. The dot products are determined and the vector is used which maximizes them. By doing so, a triple (e_1, e_2, e_3) of vectors for each triangle is obtained. The triple (e_1, e_2, e_3) and $(-e_1, -e_2, -e_3)$ is used to obtain two vector fields for each triangle by barycentric interpolation. With these two final vector fields, two different streamlines for each triangle can be generated.

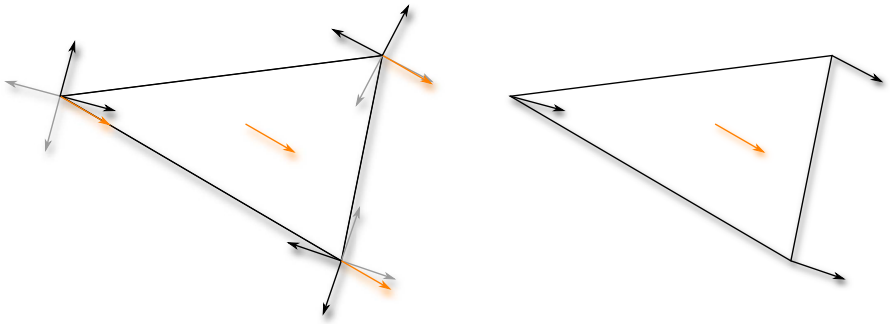


Figure 39: The maximal PCD of the triangle is compared to the PCDs per vertex. The most similar PCD of every vertex compared to the maximal PCD of the triangle is selected.

Streamline starting points: Every streamline starts at the barycenter of the contour triangle. Furthermore, streamlines are also generated at the barycenter of adjacent triangles within the *contour margin*. To achieve frame-coherence, the streamlines have to be generated at consistent start positions. The streamlines were also started at different randomly chosen start positions, but no significant visual differences occurred. Thus, the barycenter serves as the start position for the streamlines.

Streamline length: The principal curvatures are employed to determine the length of the streamline. Every triangle obtains a minimum and maximum curvature κ_1, κ_2. The length L of a streamline is calculated by:

$$L = \frac{\pi}{3 \cdot \max\{|\kappa_1|, |\kappa_2|\}}. \tag{23}$$

The intuition is to have a length of one third of the half perimeter of the osculating circle: $\frac{1}{3}\pi r$. If the streamline length L exceeds a user-defined threshold L_{max} the length is clamped: $L := L_{max}$. The median value of all calculated lengths per vertex is suggested for the streamline length threshold L_{max}.

Calculation of the streamline: Some authors use implicit or explicit iterative methods for the approximation of ordinary differential equations, e.g., [211]. Nielson and Jung [152] determined explicit streamlines over tetrahedral domains. Furthermore, they gave the solution for streamlines over triangle domains. The explicit streamline solution is derived for each triangle and this process will be described in detail. Given a triangle $t = (p_1, p_2, p_3)$ with associated PCDs (e_1, e_2, e_3), the associated (non-normalized and non-orthogonal) basis (u, v) is built by: $u = p_2 - p_1$ and $v = p_3 - p_1$. Every position p and the associated vector $e(p)$ in the interior of the triangle can be calculated by the basis (u, v), see Figure 40. The parametrized streamline

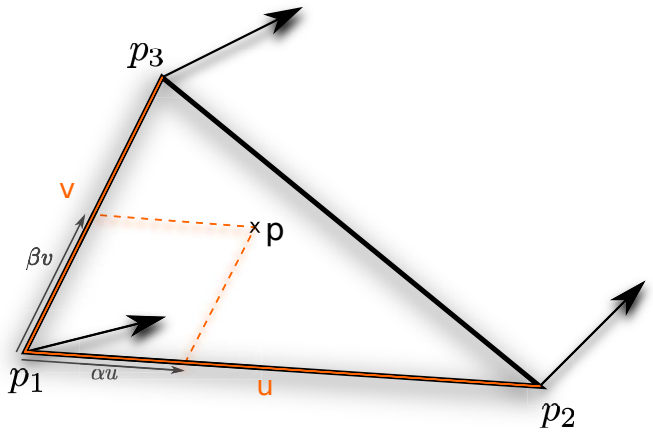


Figure 40: Every point p in the triangle can be represented by the basis (u, v) with coefficients (α, β), $\alpha, \beta \geqslant 0$ and $\alpha + \beta \leqslant 1$. The corresponding resulting vector $e(p)$ can be obtained with the point $p = (\alpha, \beta)$ by: $e(p) = (e_2 - e_1 \quad e_3 - e_1) \cdot p + e_1$.

$c(t)$ fulfills the following condition:

$$\frac{\partial c(t)}{\partial t} = \underbrace{\begin{pmatrix} e_2 - e_1 & e_3 - e_1 \end{pmatrix}}_{=:A} c(t) + e_1. \tag{24}$$

Such an inhomogeneous ordinary differential equation (iODE) $c' = Ac + e_1$ is solved by the method of separation of constants [81, 157]. First, the homogeneous ODE (hODE) $\tilde{c}' = A\tilde{c}$ is solved. Afterwards, the missing factor for the inhomogeneous part e_1 is determined. The solution of $\tilde{c}' = A\tilde{c}$ is a linear combination of a fundamental solution:

$$\tilde{c}(t) = \exp(A \cdot t) = \sum_{k=0}^{\infty} \frac{A^k \cdot t^k}{k!}.$$

In practice, the Matrix A is decomposed in a Jordan-form: $A = DJD^{-1}$. Then

$$\tilde{c}(t) = D \cdot \exp(J \cdot t) \cdot D^{-1}$$

is obtained. As the inverse is not necessary for a fundamental solution system, the fundamental system

$$\tilde{c}(t) = D \cdot \exp(J \cdot t)$$

is obtained. The solution leads to the general solution of the ODE: $c(t) = c_p(t) + \tilde{c}(t)$. The equation: $c_p(t) = \tilde{c}(t) \cdot f(t)$ is used with an arbitrary 2D differentiable function $f(t)$ to determine a particular solution $c_p(t)$ for the iODE. The derivative of $c_p(t)$ yields:

$$\frac{\partial c_p(t)}{\partial t} = A \cdot c(t) + \tilde{c}(t) \cdot f'(t). \tag{25}$$

Comparing Equation 24 with Equation 25 leads to: $\tilde{c}(t) \cdot f'(t) = e_1$. Finally, the particular solution has the form:

$$c_p(t) = D \cdot \exp(J \cdot t) \cdot \left[\int_{t_0}^{t} \exp(-J \cdot x)\, dx \right] \cdot D^{-1} \cdot e_1 + C. \tag{26}$$

For simplicity, $c_p(t)$ is written as an indefinite integral and the constant C is left out:

$$c_p(t) = D \cdot \exp(J \cdot t) \cdot \left[\int \exp(-J \cdot t)\, dt \right] \cdot D^{-1} \cdot e_1.$$

Furthermore, $\mathcal{A}(t)$ is defined:

$$\mathcal{A}(t) := D \cdot \exp(J \cdot t) \cdot \left[\int \exp(-J \cdot t)\, dt \right] \cdot D^{-1}. \tag{27}$$

The final fundamental solution for the ODE is obtained. For the specific solution, the parameters $x_1, x_2 \in \mathbb{R}$ for the hODE are needed. With the restriction $c(0) = p$, the condition: $p = \mathcal{A}(0) \cdot e_1 + D \cdot \binom{x_1}{x_2}$ is obtained. Solving this equation with respect to x_1, x_2 yields:

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = D^{-1} \cdot (p - \mathcal{A}(0) \cdot e_1). \tag{28}$$

Finally, the explicit streamline representation is obtained with starting at position p with Equation 26 and 28:

$$c(t) = c_p(t) + \tilde{c}(t) \cdot \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \tag{29}$$

$$= \mathcal{A}(t) \cdot e_1 + D \cdot \exp(J \cdot t) \cdot D^{-1} \cdot (p - \mathcal{A}(0) \cdot e_1). \tag{30}$$

The streamline representation depends on the terms $\exp(J \cdot t)$ and $\mathcal{A}(t)$, which can be simplified. The simplification will be presented in the following. The Jordan-matrix J can be represented by different forms. In order to simplify $\exp(J \cdot t)$ and $\int \exp(-J \cdot t)\, dt$ all cases of J have to be considered:

i. $J = \begin{pmatrix} \kappa & 1 \\ 0 & \kappa \end{pmatrix}$, $\kappa \in \mathbb{R}$.

ii. $J = \begin{pmatrix} \kappa_1 & 0 \\ 0 & \kappa_2 \end{pmatrix}$, $\kappa_1, \kappa_2 \in \mathbb{C}$.

Lemma 5.2.2. *For one of the cases i. or ii. different simplified representations for* $\exp(J \cdot t)$ *are obtained:*

a) i. & $\kappa \neq 0$: $\exp(J \cdot t) = \exp(\kappa t) \begin{pmatrix} 1 & t \\ 0 & 1 \end{pmatrix}$,

b) i. & $\kappa = 0$: $\exp(J \cdot t) = \begin{pmatrix} 1 & t \\ 0 & 1 \end{pmatrix}$,

c) ii. : $\exp(J \cdot t) = \begin{pmatrix} \exp(t \cdot \kappa_1) & 0 \\ 0 & \exp(t \cdot \kappa_2) \end{pmatrix}$.

Proof. **a)** First, using the definition of the matrix exponential:

$$\exp(J \cdot t) = \sum_{k=0}^{\infty} \frac{J^k t^k}{k!}.$$

Applying induction leads to

$$J^n = \begin{pmatrix} \kappa^n & n \cdot \kappa^{n-1} \\ 0 & \kappa^n \end{pmatrix}.$$

This yields

$$\exp(J \cdot t) = \exp(\kappa t) \begin{pmatrix} 1 & t \\ 0 & 1 \end{pmatrix}.$$

b) J is a nilpotent matrix therefore: $J^0 = \mathrm{Id}$, $J^1 = J$, and $J^2 = 0$. This yields

$$\exp(J \cdot t) = \begin{pmatrix} 1 & t \\ 0 & 1 \end{pmatrix}.$$

c) This case can be verified by knowing

$$J^n = \begin{pmatrix} \kappa_1^n & 0 \\ 0 & \kappa_2^n \end{pmatrix}$$

and this leads to

$$\exp(J \cdot t) = \begin{pmatrix} \exp(t \cdot \kappa_1) & 0 \\ 0 & \exp(t \cdot \kappa_2) \end{pmatrix}.$$

□

Lemma 5.2.3. *For one of the cases i., ii. or iii. different simplified representations for* $\mathcal{A}(t) := D \cdot \exp(J \cdot t) \cdot \left[\int \exp(-J \cdot t)\, dt\right] \cdot D^{-1}$ *are given:*

a) i. & $\kappa \neq 0$*:*

$$\mathcal{A}(t) = -D \cdot \begin{pmatrix} -\frac{1}{\kappa} & \frac{1}{\kappa^2} \\ 0 & -\frac{1}{\kappa} \end{pmatrix} \cdot D^{-1}$$

b) i. & $\kappa = 0$*:*

$$\mathcal{A}(t) = D \cdot \begin{pmatrix} t & -\frac{t^2}{2} \\ 0 & t \end{pmatrix} \cdot D^{-1}$$

c) ii. & $\kappa_1, \kappa_2 \neq 0$*:*

$$\mathcal{A}(t) = D \cdot \begin{pmatrix} -\frac{1}{\kappa_1} & 0 \\ 0 & -\frac{1}{\kappa_2} \end{pmatrix} \cdot D^{-1}$$

d) ii. & $\kappa_1 \neq 0, \kappa_2 = 0$:

$$\mathcal{A}(t) = D \cdot \begin{pmatrix} -\frac{1}{\kappa_1} & 0 \\ 0 & t \end{pmatrix} \cdot D^{-1}$$

e) iii.

$$\mathcal{A}(t) = \Re(\kappa^{-1}) \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \frac{\Im(\kappa^{-1})}{bc - ad} \begin{pmatrix} ac + bd & -a^2 - b^2 \\ c^2 + d^2 & -ac - bd \end{pmatrix},$$

$$\textit{with } D = \begin{pmatrix} a + bi & a - bi \\ c + di & c - di \end{pmatrix}.$$

Proof. **a)-d)** The matrix exponential has the property: $(\exp(A))^{-1} = \exp(-A)$. By using the results from Lemma 5.2.2 and inverting the matrix exponential and integrating the result, the simplification from a)-d) are obtained.
e) First, when having a complex eigenvector v with associated eigenvalue κ then the complex conjugated eigenvector $\bar{v}$ is also an eigenvector with the corresponding complex conjugated eigenvalue $\bar{\kappa}$. This can be shown by assuming that a matrix A has a complex eigenvector v with associated eigenvalue κ writing $Av = \kappa v$ and rearranging this term in the real and complex part: $A\Re(v) + iA\Im(v) = (\Re(\kappa) + i\Im(\kappa)) \cdot (Re(v) + i\Im(v))$. Comparing the real and the complex part leads to the proof that changing the sign of the complex part of the eigenvalue and the eigenvector will clear the change of signs on both sides of the equation. This proofs that D can be represented with

$$D = \begin{pmatrix} a + bi & a - bi \\ c + di & c - di \end{pmatrix}$$

and

$$J = \begin{pmatrix} \kappa & 0 \\ 0 & \bar{\kappa} \end{pmatrix}.$$

Case c) gives

$$\mathcal{J} := \exp(J \cdot t) \cdot \left[\int \exp(-J \cdot t)\, dt \right] = \begin{pmatrix} -\frac{1}{\kappa} & 0 \\ 0 & -\frac{1}{\bar{\kappa}} \end{pmatrix}.$$

Considering the real and the complex part of $\mathcal{J}$ yields:

$$\begin{aligned}
\mathcal{A}(t) &= D(\Re(\mathcal{J}) + i\Im(\mathcal{J}))D^{-1} \\
&= D\Re(\mathcal{J})D^{-1} + iD\Im(\mathcal{J})D \\
&= \Re(\mathcal{J})Id + i\Im(\mathcal{J})D \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} D^{-1} \\
&= \Re(\mathcal{J})Id + i\Im(\mathcal{J}) \begin{pmatrix} a+bi & a-bi \\ c+di & c-di \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \\
&\qquad \cdot \frac{1}{2i(bc-ad)} \begin{pmatrix} c-di & -a+bi \\ -c-di & a+bi \end{pmatrix} \\
&= \Re(\mathcal{J})Id + i\Im(\mathcal{J}) \begin{pmatrix} a+bi & a-bi \\ c+di & c-di \end{pmatrix} \frac{1}{2i(bc-ad)} \\
&\qquad \cdot \begin{pmatrix} c-di & -a+bi \\ c+di & -a-bi \end{pmatrix} \\
&= \Re(\mathcal{J})Id + i\Im(\mathcal{J}) \frac{1}{2i(bc-ad)} \begin{pmatrix} 2(ac+bd) & 2(-a^2-b^2) \\ 2(c^2+d^2) & -2(ac+bd) \end{pmatrix} \\
&= \Re(\mathcal{J})Id + \Im(\mathcal{J}) \frac{1}{(bc-ad)} \begin{pmatrix} ac+bd & -a^2-b^2 \\ c^2+d^2 & -ac-bd \end{pmatrix}
\end{aligned}$$

This proofs e). □

Within the triangle, streamlines have the following properties:

$$c(t)_x \geqslant 0, c(t)_y \geqslant 0,$$

and

$$c(t)_x + c(t)_y \leqslant 1.$$

Violating one condition leads to an intersection point of the streamline with one of the edges. Thus, the adjacent triangle of the edge is obtained. The underlying vector field of the new triangle can be determined. Instead of using the triangle PCD as reference direction, the streamline direction vector is employed.

Adaptive step size: For streamline propagation, an adaptive step size is used. Whenever the length of a streamline segment exceeds the inradius of the triangle, the step size has to be halved. This ensures that the streamline will not immediately leave the triangle in most cases. Furthermore, the visual results of the streamlines seem to be smooth, although not too much line segments in a triangle have to be calculated.

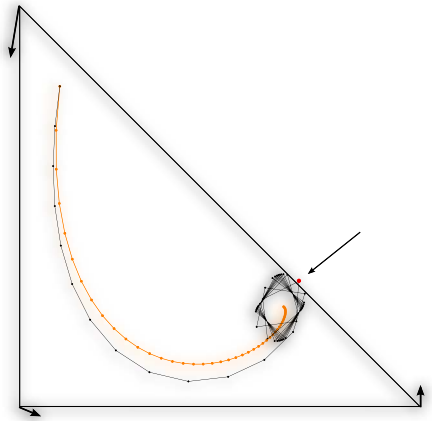


Figure 41: A comparison between explicit and implicit streamline computation using the forward Euler method. On the one hand, an increasing error during the computation is obtained. On the other hand, the implicit streamline may cause wrong results. The explicit streamline approximates the singularity. Whereas the implicit streamline oscillates around the singularity and can leave the triangle in the worst case.

5.2.4 *Advantage of explicit streamline calculation*

First of all, determining the explicit streamline exhibits a lower error than iterative methods for the approximation of ODEs. Explicit streamlines do only produce an error when approximating the intersection point of one edge of the triangle with the streamline. Another advantage of explicit streamlines is the converging propagation behavior towards singularities. The explicit streamline will converge into this point. An iterative streamline can oscillate around the singularity (see Fig. 41). Moreover, iterative methods are based on a sequence of points. Thus, one point can only be approximated by using the previously calculated ones. In contrast, explicit streamlines can be calculated in parallel without knowing the previous points.

5.3 ALGORITHM AND GPU IMPLEMENTATION

The algorithm for ConFIS on the mesh M is as follows:

1. (Optionally) Subdivide and smooth M.
2. Compute the PCDs and the corresponding PCs.
3. Determine feature regions of M.
4. Compute two streamlines for all triangles.
5. Compute the contour and *contour margin*.
6. Draw the contour and feature streamlines.

The algorithm is divided into two parts. The first part (1.-4.), consists of the preprocessing steps and the second part (5.-6.) is executed during runtime (see Fig. 43). As shown earlier, the generation of a streamline for an arbitrary triangle requires to traverse M iteratively from one triangle to the next. For achieving a fast rendering, several APIs, such as CUDA, OpenCL, or DirectCompute are available. All computations are performed with the OpenGL shader framework to be independent of graphics card vendors and to reduce any overhead by additional APIs. The shader concept is ideally suited for per-vertex and per-triangle operations, which are required by the ConFIS method. OpenGL shaders natively do not provide neighborhood information, such as the 1-ring of each vertex. Thus, a data structure similar to the one which has been presented in [227] is employed. Topological information, such as the location of neighboring vertices, is made available via the OpenGL extension ARB_SHADER_STORAGE_BUFFER_OBJECT, which is part of the OpenGL core since version 4.3. First, this extension is used to build neighbor information. A 3D integer vector N is created. The x-component consists of the ordered neighbors of the consecutive vertices. This means, the ordered neighbors of the first vertex are written clockwise (or counter-clockwise) in a row. Next, the ordered neighbors of the second vertex are used and they are stored behind the first neighbors. So, the x-component consists of

$$N_x = (\ \mathrm{star}_{\mathrm{Ord}}(0),\ \mathrm{star}_{\mathrm{Ord}}(1),\ \mathrm{star}_{\mathrm{Ord}}(2), \dots\ ),$$

where $\mathrm{star}_{\mathrm{Ord}}(i)$ are the ordered neighbors of the ith vertex. The second component stores the number of neighbors of each vertex. Thus, the y-component yields:

$$N_y = \{|\mathrm{star}_{\mathrm{Ord}}(0)|,\ |\mathrm{star}_{\mathrm{Ord}}(1)|,\ |\mathrm{star}_{\mathrm{Ord}}(2)|, \dots\}.$$

The z-component is the offset index for the corresponding entry in the x-component:

$$N_z = \{0, |\mathrm{star}_{\mathrm{Ord}}(0)|,\ |\mathrm{star}_{\mathrm{Ord}}(0)| + |\mathrm{star}_{\mathrm{Ord}}(1)|, \dots\}.$$

See Figure 42 for an illustration of the neighbor structure. Having the current vertex at index i, the number of neighbors and the indices of these neighbors can be immediately accessed. For the number of neighbors, the entry $\mathrm{num} = N_y(i)$ is needed. For the assessment of the neighbors, the offset $\mathrm{off} = N_z(i)$ is needed and it yields the neighbors:

$$\{N_z(\mathrm{off}), N_z(\mathrm{off}+1), \dots, N_z(\mathrm{off}+\mathrm{num}-1)\}.$$

With this, a streamline can be seeded and tracked at each triangle for an arbitrary number of triangles.

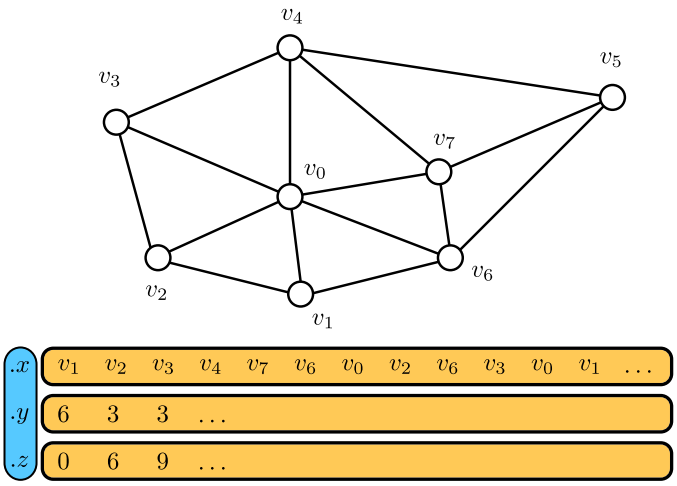


Figure 42: Neighbor structure of a triangle mesh using the storage buffer.

5.3.1 *Preprocessing*

First, two streamlines are generated at the barycenter for each triangle with given length. These processing steps are entirely executed on the GPU via OpenGL shaders. Such a streamline buffer contains #Triangles $\times$ #LineSegments $\times$ 2 elements. For writing the streamline vertices to this buffer, shader storage buffer objects are used. Thus, the OpenGL extension SHADER_STORAGE_BUFFER_OBJECT, which is part of the OpenGL core since version 4.3, is required. With this, each triangle is able to write the vertices belonging to each of its two streamlines into the streamline buffer. Additionally, the preprocessing step allows for a detection of those triangles which shall later generate a feature streamline based on the curvature criterion. Until this point, nothing has been drawn – but all the streamlines have been precomputed for later rendering.

5.3.2 *Rendering Loop*

During runtime, the steps 5 and 6 are executed. Using geometry shaders, the object contour and a defined *contour margin* can be determined. Thus, the previously generated streamline buffer can be accessed and the contour and feature streamlines can be drawn. Unfortunately, this yields a bad load balancing, since streamlines are seeded only at comparably few triangles. Most geometry shader invocations would not perform any streamline processing. In the worst case, all threads in one thread group have to wait until all streamline-generation-threads finish – even if only one thread is generating a streamline.

Thus, the rendering stage is split into two render passes. At first, all triangles, which shall draw a contour streamline, are identified and marked. For each of these triangles and those which have been

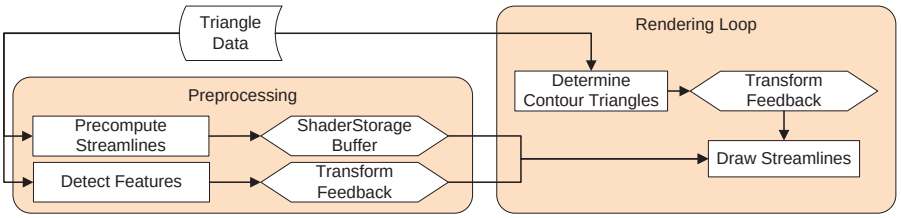


Figure 43: The scheme of ConFIS which is divided in two parts: the preprocessing and the rendering part.

marked during preprocessing for seeding feature streamlines, the triangle information (e.g., vertex IDs) are stored using OpenGL's transform feedback mechanism. In the seconds pass, rendering is performed only for the marked triangles. As a result, the GPU processes only triangles which contribute to streamline rendering. Each of these triangles can access the complete streamline buffer via TEXTURE_BUFFER_OBJECTS (TBOs). The opacity of each streamline is reduced with increasing length.

5.4 EVALUATION

Two different evaluations are conducted to assess the capabilities of the ConFIS method. First, a qualitative evaluation for the five line drawing techniques: *suggestive contour* (SC), *apparent ridges* (AR), *photic extremum lines* (PEL, according to Zhang et al. [212]), *high quality hatching* (HQ), and ConFIS is performed. This evaluation leaves out the feature line methods ridges and valleys, demarcating curves, and Laplacian lines. As seen in Figure 32, the three omitted feature line methods are similar to the compared feature lines. Furthermore, ridges and valleys are not view-dependent, demarcating curves are very susceptible to noise, and Laplacian Lines need to compute the Belkin weights of the Laplace operator, which is very unintuitive for medical experts.

The second evaluation compares the ConFIS method with hatching methods on endoscopic views.

For the first evaluation, the goal was to assess their capabilities for expressing relevant surface characteristics. The purpose was to figure out, which of the line drawing methods yields the most expressive result for the participants. The evaluation was conducted with two physicians and three researchers with background in medical visualization. Four representative surface models were chosen: ribs (Fig. 48a), aneurysm 1 (Fig. 44, middle row), trachea (Fig. 44, bottom row), and femur (Fig. 45d). The models are derived from segmenting medical image data and preprocessed to ensure an appropriate and homogeneous degree of tessellation. For all compared methods, the original implementations by the corresponding authors are used. The evaluation took about 40–60 minutes and was conducted in three tasks:

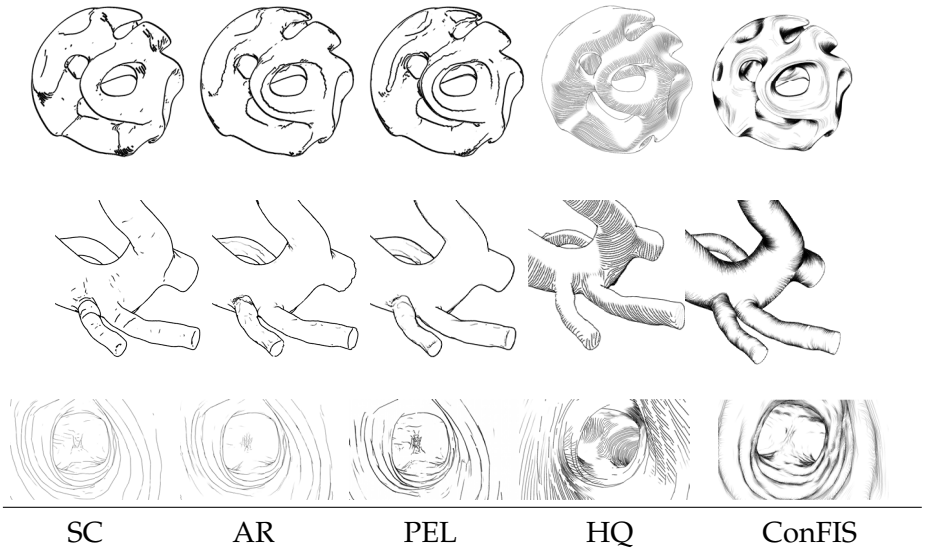


Figure 44: Different surface models displayed with *suggestive contour* (SC), *apparent ridges* (AR), *photic extremum lines* (PEL), *high quality hatching* (HQ), and ConFIS. The models are from top to bottom: hyperthing, aneurysm 1, and endoscopic view of a trachea.

1. Each participant was shown the shaded surface models in a different order. They had the possibility to interactively explore the model and gain a 3D impression.

2. The second task was to adjust the specific parameters of the illustration methods to obtain a subjectively satisfying and informative result. During this, the participants' spoken comments and the parameter sets were noted.

3. The third task of the evaluation consisted of a visual comparison and a qualitative assessment between the feature line methods. Based on the recorded parameter sets, each participant should assess which method is more appropriate to express surface features and which limitations have been observed.

Aneurysm 1 model (Fig. 44): The participants mentioned that the lines generated by the feature line methods were not appropriate to gain a comprehensive spatial impression. Most of the generated lines were considered to be distracting. On the other hand, these methods depicted parts of the bifurcation well. All participants agreed that the hatching method generates a reasonable 3D impression. For HQ, the evenly spread lines cannot depict important features, e.g., the border between the vessel and aneurysm sac. Furthermore, several participants criticized the low performance of the HQ implementation (~13 fps). ConFIS fulfilled the demands to illustrate relevant features and to convey an appropriate 3D impression. All participants chose ConFIS as their preferred line drawing technique.

Trachea model (Fig. 44): The inner view of the trachea has two features: the elongated structure and the bifurcation where the carina tracheae splits into both branches. The participants confirmed that the feature line methods can depict the elongated structures, but fail to display the carina tracheae. Apart from that, they explained that the hatching method as well as ConFIS depict both properties well. One participant found some streamlines slightly disturbing and unnecessary to gain a spatial impression. The hatching method could not highlight the bifurcation features. In contrast, it looked like a planar transition from one bronchus to the other. Finally, all participants preferred the ConFIS method.

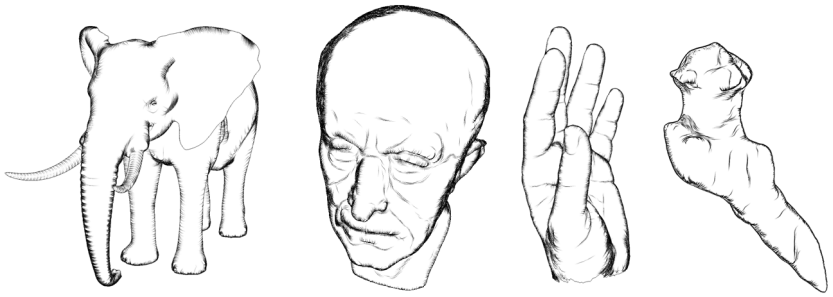
Ribs model (Fig. 48a): The ribs model was chosen to evaluate the 3D feeling of the mesh even if the surface has a lot of structures. The participants get a reasonable 3D impression by all line drawing methods. Some participants mentioned that there are only small differences between the feature line methods. One participant explained that the impression of the model is appropriate during the interaction, but seeing only an image would be confusing. The participant could not set the ribs apart from the gaps. Furthermore, some lines which are produced by feature line methods are distracting and the hatching method cannot illustrate the dents. ConFIS illustrates all ribs well and the participants can distinguish the ribs from the gaps and all dents are depicted as well.

Femur model (Fig. 45d): Some of the participants found fault with the view-dependent feature illustrations. The feature line methods only show some dents first if the camera position is chosen well. Two participants criticized the missing details using the hatching method. Some dents are missing and without interactive exploration some important regions are missed. Those regions have been highlighted with ConFIS, which was again preferred.

5.4.1 *Results*

The results of the evaluation can be summarized as follows:

- Current line drawing techniques achieve satisfying results only if the models exhibit a smooth and regularly tessellated surface.
- The clutter of surfaces derived from measured image data, such as noise and staircase artifacts, are usually emphasized.
- For some cases, line drawing methods are not able to depict relevant features.



(a) Elephant (b) Max Planck (c) Hand (d) Femur

Figure 45: The ConFIS method applied to different surface models.

- The ConFIS method was the most expressive technique in the comparison.

However, the informal study does not allow a definitive statement and requires further evaluation. ConFIS is able to provide a sparse representation of a model's surface, since illustrative patterns are drawn along characteristic contours and only sparsely within the surface. Thus, ConFIS might also serve to depict the anatomical context as spatial reference in medical illustrations, e.g., flow visualization. ConFIS provides such spatial information and gives also hints on local shape properties. However, the discussion with the physicians showed that such illustrations are not suitable for diagnostic purposes. In therapy planning, illustrative pictures are used for discussions. Especially in neck surgery, physicians use abstract 3D illustrations as a printout to draw resection lines and access path planning. In the following it will be shown that ConFIS gives also visual pleasing results in endoscopic views. First, it will be shown that feature lines are not appropriate for endoscopic views and second a qualitative evaluation is conducted where ConFIS will be compared to the high-quality hatching and to the real-time hatching method by Praun et al. [163].

5.4.2 *Endoscopic Data*

Clinical image data such as computed tomography (CT) or magnetic resonance angiography (MRA) are used to acquire the anatomical information as well as the surface model. The surface mesh is reconstructed by applying a simple thresholding segmentation followed by a connected component analysis. The resulting segmentation mask is used to construct the surface by a marching cubes algorithm. Afterwards, the mesh quality is improved by a combination of edge collapses, edge flips, smoothing, and remeshing.

(a) Shading (b) RV (c) SC (d) AR (e) PEL

Figure 46: Endoscopic view rendered with simple shading (a), *ridges and valleys* (RV) (b), *suggestive contours* (SC) (c), *apparant ridges* (AR) (d), and *photic extremum lines* (PEL) (e).

5.4.3 *Hatching*

HATCHING: First, Figure 46 illustrates that feature line methods are not appropriate for endoscopic views. For the comparison of hatching methods, three different techniques are considered: the real-time hatching by Praun et al. [163], the high-quality hatching by Zander et al. [211], and ConFIS. Praun et al. introduced *real-time hatching*. They generate line-art tonal art maps for different shading levels. Afterwards, lapped textures are applied to map the line-art textures onto the surface. Thereby, the textures are mapped randomly onto the surface and missing facets are processed by querying a list of non-covered facets. Then, those textures are used which correspond to the underlying shading. Zander et al. employ *high-quality hatching*, a geometry-based method. They do not use textures, but streamlines. These streamlines are generated on the entire surface and propagated along the principle curvature directions. The shading of each streamline part corresponds to the underlying surface shading.

5.4.4 *Evaluation of Hatching on Endoscopic Views*

A qualitative evaluation of the three line drawing techniques was performed to rate the ability for assessing the spatial impression. For the evaluation, four surface models were chosen. The evaluation was conducted with seven researchers who are familiar with medical visualizations. The illustrations were generated and the results were shown in different order. The sequence of the data sets was the same. The ordering of the line drawing techniques changed. First, the researchers were shown different results and they were asked if they are able to perceive branches and other features from the resulting pictures. After all illustrative pictures were shown, the normal shaded images were presented. Then, they were asked if they would have expected this model. Afterwards, all results were shown to compare between the different methods. Here, the goal was to figure out if some features were misinterpreted or missed. In comparison of all line drawing methods with the shaded model, the participants should rate which technique is more appropriate to capture salient regions

as well as the spatiality and which limitations they noticed. During the evaluation, the spoken comments of the participants were noted.

5.4.5 *Results*

In Figure 47, the shaded models are shown. For each model, the results of the different hatching methods are depicted. The results of the evaluation are summarized in Table 4.

Table 4: The left table shows which method was rated as most effective for the four data sets (left column). The participants could also vote for two techniques (maximum is seven). The right table shows how many participants counted the right number of branches.

	RT	HQ	ConFIS
1	4	0	6
2	2	0	6
3	5	0	2
4	1	0	7

	RT	HQ	ConFIS
1	5	5	7
2	5	7	7
3	7	7	7
4	6	6	6

Two findings were assessed. First, which technique was rated as most expressive and second, which method delivers a good result for perceiving the right number of branches. The spoken comments of the participants were very similar. The *real-time hatching* (*RT*) gives a good spatial impression, whereas the *high-quality hatching* (*HQ*) is insufficient for a 3D impression. The *ConFIS* is also able to deliver a spatial impression and tries to depict salient regions. Especially for the third model, the cut-off between the two branches was illustrated appropriately. In contrast, the *HQ* could not depict it well. Furthermore, the participants liked the consistent drawing of the *ConFIS* method. Whenever a branch is depicted, the lines wrap around the outgoing structure. The *RT* technique is able to illustrate the model according to the light intensity. Therefore, the result is very close according to the shaded image. Two participants realized some distortions of the *RT* method as this is a texture projection method. The method was implemented but another parameterization technique was used. Nevertheless, they mentioned that these artifacts were not distracting to focus on the object branches. One participant thought of an improvement of *HQ* and *ConFIS* by attenuating the lines according to the distance to gain better depth cues. In summary, the *ConFIS* technique was rated as the most expressive technique and gives the best impression for branches.

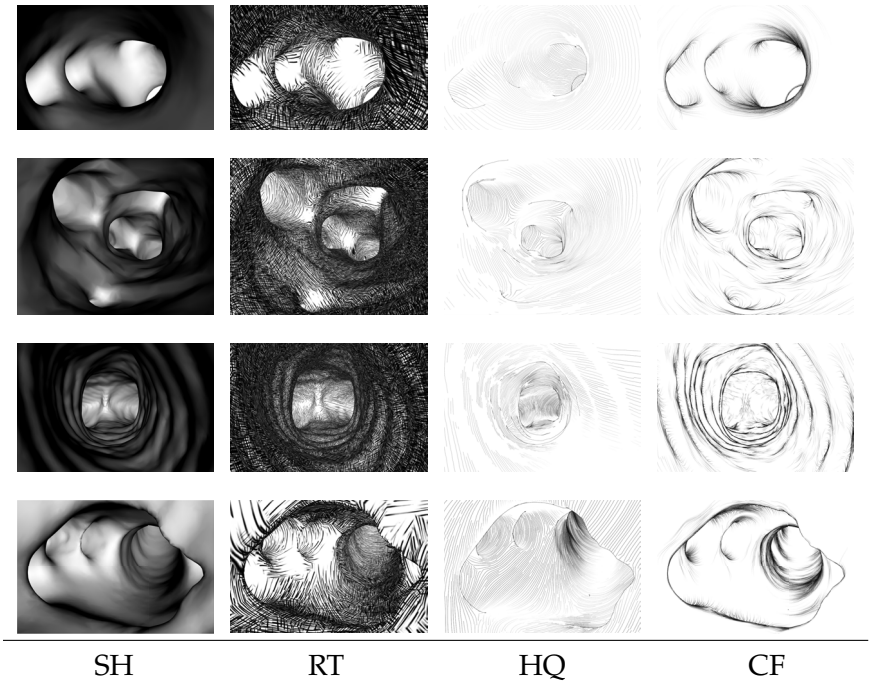
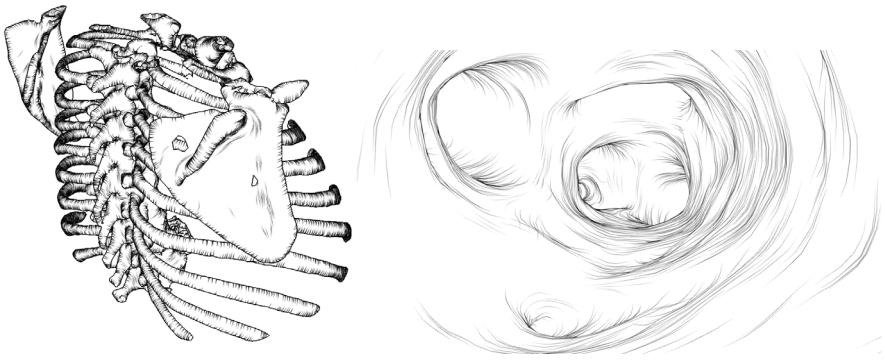


Figure 47: Endoscopic view illustrated in simple *shading* (SH), *real-time hatching* (RT), *high-quality hatching* (HQ), and *ConFIS* (CF).

5.4.6 *Discussion*

The result of the evaluation can be summarized such that all methods are able to illustrate the underlying surface model. The *RT* method uses the underlying shading to create an illustrative visualization result which is close to the shaded image. The *HQ* technique generates streamlines on the whole surface and shades them according to the underlying illumination. In comparison, the *ConFIS* technique depicts the spatiality by illustrating the contour margin as well as curvature-based features. Therefore, if the user demands a salient representation with a spatial impression, the *ConFIS* method is recommended. For a full visualization of the object and using an alternative to diffuse lightning, the *RT* method is also recommended. As the method by Praun et al. uses a projection of the texture onto the surface, it is sensitive to surface noise and the results depend strongly on the local parameterization. To avoid distortions, the mapped texture should be small to prevent a covering of a large high frequent surface. This implies a longer preprocessing time for determining the lapped textures and a result which is close to normal shading. *HQ* and *ConFIS* depend on the underlying curvature field. The presented curvature field by Rusinkiewicz is robust against surface noise. Here, the result is that the *ConFIS* method is faster than the high-quality hatching. This is explainable by the fact that *ConFIS* draws only streamlines at salient regions and therefore less streamlines than the high-quality hatching.



(a) Ribs (b) Inside view of the pulmonary artery

Figure 48: The ConFIS method with two anatomical surface models.

Surprisingly, all methods were able to illustrate noisy surfaces well. This can be seen from the second data set of the evaluation. Regarding the evaluation, it can be stated that hatching methods offer an alternative to normal shading. They can be used for context-aware medical illustrations in endoscopic views as well as learning illustrations for textbooks.

5.5 RESULTS

Different experiments were performed in order to assess the performance of ConFIS. The experiments consider the frame rate achieved with ConFIS for different artificial and anatomical surface meshes (see also Fig. 48 and Tab. 5). The approach was implemented on a mid-class desktop computer with an Intel Core i7 CPU (2.8GHz), 4GB RAM, and an NVidia GeForce GTX 660 Ti. For all employed surface models, rendering could be performed in real-time. Surface models with given curvature vector field with the corresponding number of triangles, averaged generated streamlines, initialization time, and average frame rates are summarized in Table 5. Streamline calculation and feature region detection is performed only once during the preprocessing step. The latter requires a lot of memory for storing the vertices of the precomputed streamlines. For a sample model with about 100k faces and, e.g., 50 vertices per streamline, such buffer may comprise ~153 MB. However, with recent graphics cards, at least 1024 MB are usually available.

During runtime, only the contour is determined and streamlines at contours and features are drawn. The approach depends on two user-defined values, which the evaluation participants confirmed to be intuitive. Different models are used and the results of different line drawing techniques were compared, see Figure 44. In most experiments, ConFIS could express the relevant surface features (see Fig. 48). Since the illustration technique seeds streamlines at the barycenter of triangles, it is tessellation-dependent. Thus, for low resolu-

Table 5: Performance test of ConFIS for all shown models.

Model	#Δ	# SL	Init per s	FPS
Aneurysm 2	16,778	5,134	2,735	440
Cow	23,216	7,788	0,559	260
Femur	40,978	11,734	0,697	160
Trachea	69,964	25,501	1,617	127
Portal Vein	80,062	23,067	2,849	155
Ribs	85,736	26,780	3,071	87
Hyperthing	88,756	41,023	2,270	57
Max Planck	98,260	30,407	1,992	90
Aneurysm 1	98,970	32,659	2,002	78
Pulmonary Artery	100,000	30,680	2,735	104
Hand	105,860	25,915	2,298	72
Elephant	157,160	48,875	2,665	57

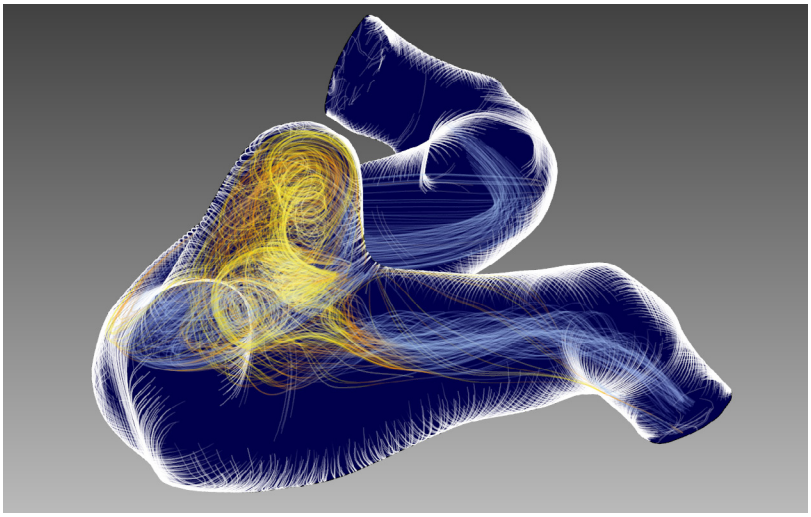


Figure 49: Fresnel alternative with white streamlines (aneurysm 2).

tion meshes, the overall visual impression will be disturbed by only sparsely drawn lines. However, the user could always get an impression of the surface characteristics. During the tests with other feature line and hatching methods, it was noticed that this seems to be a general problem.

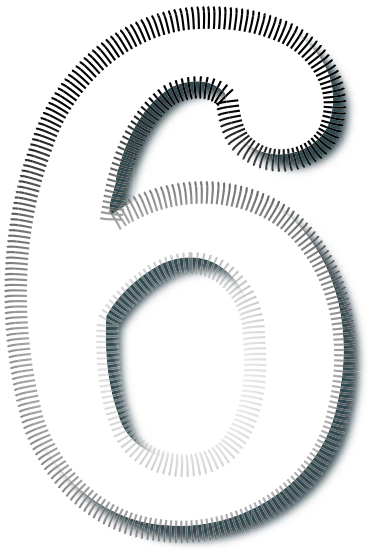
5.6 CONCLUSION

In this chapter, ConFIS was presented – a novel illustrative visualization technique for surface models based on streamlines. The streamlines have the advantage that the user gets a natural impression of

the curvature of the model. Furthermore, the user gains an enhanced 3D impression. Different comparisons to other feature line methods were made. The experiments showed that ConFIS depicts most of the surface models well and gives also a good impression of endoscopic views. ConFIS illustrates only salient regions which fulfills the conditions. Obviously, convex regions will not be illustrated. The parameters can be modified to emphasize sharp features of models. However, more parameters are required to define a lower and upper bound. The concept of the *contour margin* was used to provide a frame-coherent illustration.

As disadvantages, the ConFIS approach is strongly tessellation-dependent and requires a preprocessing step to illustrate the surface. Therefore, in the next chapter another approach will be introduced that overcomes these issues. Thus, a method will be presented where the streamlines are consistently drawn during the run-time. As an outlook, ConFIS is considered as an alternative rendering technique. ConFIS could be used in a similar manner as Fresnel shading to convey the impression of bending anatomical structures with streamlines, see Figure 49. Fresnel shading and ConFIS are used in combination to analyze the blood flow inside the vessel and to perceive the bending of the vessel. Unfortunately, using streamlines for a visual illustration method with streamlines representing the blood flow is not appropriate, since it yields visual clutter. However, with respect to the current evaluations, ConFIS provides a good approach on filling the gap between feature line and hatching methods.

Line Integral Convolution for Illustrative Molecular Visualization



This section is partly based on:

Kai Lawonn, Michael Krone, Thomas Ertl
and Bernhard Preim
Line Integral Convolution for Real-Time
Illustration of Molecular Surface Shape and
Salient Regions
Computer Graphics Forum, 33(3), pp. 181-190, 2014

6

LINE INTEGRAL CONVOLUTION FOR ILLUSTRATIVE MOLECULAR VISUALIZATION

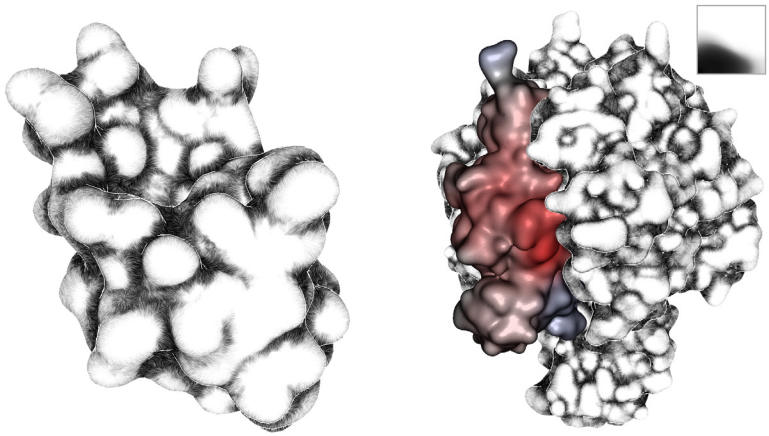
6.1 INTRODUCTION

In this chapter a novel line drawing technique is presented.This method is motivated by molecular surface renderings as shown in Figure 50, but not restricted to it. As seen in the previous chapter, the ConFIS method strongly depends on the underlying tessellation and the feature detection is based on the mean curvature. Therefore, it cannot handle animated surfaces in real-time. Especially for time-dependent molecular simulation data, an illustrative visualization technique has to fulfill several requirements. Frame-coherency is indispensable as stated in the previous chapters. Furthermore, the most important requirement is to illustrate time-varying data. Therefore, frame-coherency and real-time performance should be achieved during the deformation of the surface.

6.2 ILLUSTRATIVE VISUALIZATION FOR MOLECULAR DYNAMICS

In molecular visualization, illustrative rendering is beneficial to highlight the 3D shape and structure. Especially for biomolecules like proteins, the structure of the surface is essential for understanding function. Furthermore, the molecular visualization is important for analyzing (bio-)molecular simulations. An introduction to the most common types of biomolecules, how they work, and what effect they have in a living cell can be found in [77]. All molecular representations arise from different model definitions that depict distinct aspects of the atomic structure. The molecular models range from simple depictions of the atoms and chemical bonds to abstracted representations and molecular surfaces. The molecular surface exhibits structures (e.g. cavities, channels, and pockets) that affect the function of the protein. Therefore, an interactive visualization of the raw molecular data that highlights these structures is important for analyzing molecular dynamics. Illustrative techniques have been shown to effectively accentuate the protein structure [191]. Figure 50a shows a protein surface illustrated using our method, whereas in Figure 50b, it is applied selectively to emphasize a compartment of an ATPase.

In order to be useful for time-dependent molecular simulation data, an illustrative visualization technique has to fulfill several requirements. It should *illustrate deformable surfaces*, since the protein surface changes constantly due to conformation changes and the thermal vi-



(a) Molecular surface structure (Isomerase I)

(b) Focus-and-Context (ATPase)

Figure 50: Our method applied to two molecular surfaces: (a) illustration of the surface structure of an isomerase protein and (b) focus-and-context visualization for an ATPase. The 2D LUT shown in (b) was used for both renderings.

bration of the atoms. The molecular surface shows the boundary of a protein with respect to a certain type of solvent molecule [166]. Consequently, the surface can also change during analysis due to user input, e.g., if the user selects the surface for another type of solvent. Therefore, an *interactive performance* is necessary without time-consuming preprocessing for every animation step. Finally, for visual exploration, *frame-coherence* has to be provided during the interaction as well as during the time-dependent deformation.

This method uses line integral convolution (LIC) to assess the shape of the surface [31]. LIC is often used for the depiction of vector fields on a 2D domain. Previously, the LIC concept was employed to illustrate a vector field itself [60]. In this work, however, we use LIC to generate a hatching-like visualization for conveying the surface's shape. The illustration of the surface structure is based on the illumination gradient. Furthermore, a 2D lookup table (LUT) is used to map additional information to the surface at the respective regions. The LUT is encoded in a 2D texture that is used for color-coding (see the inset in Figure 50b). Using LIC, the results are comparable to the *ConFIS* method. However, the novel method avoids the extensive memory usage and computationally heavy preprocessing needed by *ConFIS*. Therefore, the new method can be used in real-time on animated surfaces. Disabling the LIC method, results are obtained which are similar to feature line techniques. In summary, the new illustrative visualization method makes the following contributions:

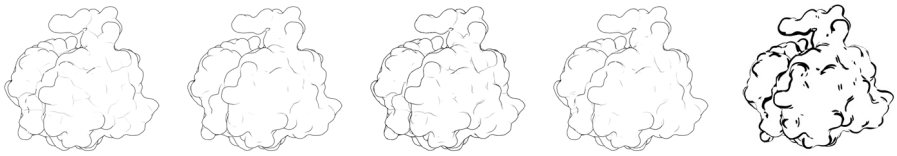
- A novel view-dependent and frame-coherent illustrative visualization method.

- Line drawing illustration based on a 2D lookup table (LUT).
- LIC is applied to salient regions in real-time.
- A method that acts on animated deforming surfaces.

Related Work for Molecular Visualization

In the following, a small overview on molecular visualization will be given. Illustrative representations have a long tradition in molecular graphics and are important for analyzing (bio-)molecular simulations. An introduction to the most common types of biomolecules, how they work and what effect they have in a living cell can be found in [77]. All molecular representations arise from different model definitions that depict distinct aspects of the atomic structure. The molecular models range from simple depictions of the atoms and chemical bonds to abstracted representations and molecular surfaces.

Goodsell and Olson [76] explained different techniques to illustrate molecular surfaces using hatching. They used 2D image processing techniques from z-buffer images. The result consists of silhouettes and hatched shading. Duncan and Olson [55] computed texture coordinates for molecular surfaces to map additional information to the surface. More recently, Tarini et al. [191] used ambient occlusion, which was first introduced by Zhukov et al. [215], and edge cueing for molecular visualization. They enhanced the perception of the molecules in employing precomputed ambient occlusion, depth-dependent contours, and halos. Contour lines were also used by Sigg et al. [181] and Lampe et al. [119] to illustrate atomistic data. Weber generated interactive pen-and-ink renderings of the so-called cartoon model (an abstracted representation of the internal protein structure) [206]. Krone et al. [112] used a GPU ray casting technique for the visualization of the smooth molecular surface of proteins. Their approach yields interactive frame rates for large data sets. Following Tarini et al. [191], they enhanced shape perception by applying contour lines and depth darkening [130], which renders depth-dependent halos to mimic screen-space ambient occlusion. Falk et al. [61] developed a framework for exploring data from simulations in a virtual cellular environment. They employed a schematic visualization abstract. Furthermore, focus-and-context techniques were used as well combining depth cues and depth of field. Van der Zwan et al. [199] introduced a method for the continuous abstraction of proteins that computes a transition between an atomistic model and the cartoon model. They used illustrative rendering styles (cel shading with ambient occlusion, hatching). An efficient algorithm for the visualization of large dynamic particle data sets was introduced by Krone et al. [113]. Parulek et al. [158] proposed a visualization technique based on the level-of-detail concept. Therefore, different visualization



(a) RV (b) SC (c) AR (d) PEL (e) LIC

Figure 51: A lipase model with *ridge and valleys* (RV), *suggestive contours* (SC), *apparent ridges* (AR), *photic extremum line* (PEL), and the LIC approach. Note that the LIC method marks salient region whereas feature lines depict salient details using lines.

methods are applied depending on the level-of-detail. Falk et al. [62] provided an atomistic visualization method for data sets comprising billions of atoms. They also employ methods to enhance the shape and depth perception. The latter two visualization techniques resemble the artistic renderings of Goodsell [77] used for the "Molecule of the Month" presented by the Protein Data Bank [15]. These renderings use cel shading and stylized ambient occlusion to obtain a clear and comprehensive, yet visually pleasing, depiction of proteins. An overview of modern molecular visualization techniques is given in [37, 128, 153].

6.3 METHOD

In the field of molecular surface visualization, important structures, e.g., bumps and dents, need to be enhanced. Therefore, these salient regions are determined and depicted by using a feature field, which is represented by a scalar field. This scalar field is determined using an illumination-based vector field. The illumination-based vector field is well-suited for conveying discontinuities in visibility and illumination. Finally, this vector field is used for calculating LIC. Therefore, the LIC approach is divided into three steps:

1. Feature vector field: An illumination-based vector field is provided.
2. Feature region: A scalar field is determined that represents salient regions based on the feature vector field.
   Optionally: A second scalar field is determined.
3. Line integral convolution: LIC is adapted to the salient regions along the vector field of step 1.

6.3.1 *Feature Vector Field*

The illumination gradient is used for the feature vector field. First, the diffuse illumination l_i of every vertex i is determined: $l_i = \langle \mathbf{n}_i, \mathbf{v}_i \rangle$.

Remark, that $\langle .,. \rangle$ denotes the dot product and $\mathbf{v}_i$ denotes the normalized view vector from $\mathbf{p}_i$ to the camera. Afterwards, the illumination gradient per triangle $\triangle = (i, j, k)$ is calculated by:

$$\nabla l_{\triangle} = (l_j - l_i)\frac{(\mathbf{p}_i - \mathbf{p}_k)^{\perp}}{2A_{\triangle}} + (l_k - l_i)\frac{(\mathbf{p}_j - \mathbf{p}_i)^{\perp}}{2A_{\triangle}}, \tag{31}$$

where $A_{\triangle}$ denotes the area of the triangle and $\perp$ stands for a counterclockwise rotation by 90° in the triangle plane, see [21]. Analogously, the illumination gradient can be calculated by Equation 15. Afterwards, the gradient ∇l_i is computed per vertex i by rotating the gradients of the adjacent triangles to the tangent space first, and then weighting them by their Voronoi area [140]. The feature field is used for feature region detection and for LIC generation.

6.3.2 *Feature Region*

A LUT is used to encode information on the surface. This is inspired by transfer functions where scalar values are assigned to colors. The LUT, like the example in 50b, was manually created and loaded as an image. Every vertex is assigned to one (optionally two) scalar values and this vertex obtains the color which corresponds to a certain position in the LUT. Therefore, two scalar fields are determined. The first scalar field φ represents salient regions on molecular surfaces, whereas the second scalar field τ is used to depict the depth perception of the shape.

The illustration of salient regions was focused on feature line methods. Due to the underlying atomistic data, the molecular surface is very uneven, i.e. it exhibits many bumps and valleys. In molecular visualization, it is essential to depict regions of high saliency such as the transition of concave and convex regions. Kolomenkin et al. [111] tried to depict the transition by determining the isolines where the change of the curvature is maximal. As this calculation needs higher order derivatives, it is sensitive to noise and, therefore, not well-suited for noisy meshes. Thus, the LIC method is focused on a view-dependent approach based on the *suggestive contours*, which illustrates the transition as well, compare Section 4.6. The computation of the feature regions is based on the previously determined illumination gradient per vertex. The first scalar field φ is determined by:

$$\varphi_i = D_{w_i} l_i = \langle \nabla l_i, w_i \rangle, \tag{32}$$

where D is the directional derivative and w_i denotes the projected view vector onto the tangent plane of $\mathbf{p}_i$. Colloquially spoken, this scalar field is zero at the inflection points representing ridges and valleys. The y-dimension of the LUT is assigned by applying $y = 1 - \alpha \cdot |\varphi|$, where α is a user-defined value, which controls the feature intensity margin. The higher α, the larger the feature region. The

scalar field $y = 1 - \alpha \cdot |\varphi|$ emphasizes the shape and enhances bumps and dents and is, thus, ideally suited for atomistic data depiction. Figure 51 shows a comparison of the feature region in comparison with feature line methods.

A second scalar field τ is used for the x-dimension of the LUT. Ambient occlusion is chosen because the shape of the molecular surface should be emphasized. Therefore, an ambient occlusion value for supporting the depth perception of the shape is needed. That is, τ is an ambient occlusion value in $[0, 1]$. Ambient occlusion can be described as follows: For each point p in the scene, the ambient occlusion value A_p is determined by the amount of ambient light that is blocked by surrounding geometry. A_p can be computed by integrating over the visible hemisphere hS using equation 33:

$$A_p = \iint_{x \in hS^2} \rho\left(L\left(p, x\right)\right) \cos\alpha \, dx, \tag{33}$$

where $\rho\left(L\left(p, x\right)\right)$ gives the amount of blocked incoming light energy from direction x for point p. $L(p, x)$ is the distance to the nearest object in the direction of x. Note that $\rho(L)$ only takes distances into account that are within a certain interval, that is, only a local neighborhood affects A_p. α is the angle between direction x and the surface normal n_p, that is, $\cos\alpha = n_p \cdot x$ represents the incoming light. There are several methods to approximate ambient occlusion in real-time. For arbitrary, complex scenes there is a variety of fast image-space methods, which are often referred to as screen space ambient occlusion. The basic idea is to sample neighboring fragments and determine the ambient occlusion based on their depth values. The screen space ambient occlusion method is used presented by Kajalin [102] for general data. As shown by [191], ambient occlusion facilitates the perception of atomistic data. Thus, this technique is employed to illustrate dents and for supporting biochemists to distinguish between dents and bumps.

Finally, both scalar fields are used that yield the coordinates of a 2D LUT represented by a texture. In Figure 52, the coordinates of $(\tau_i, 1 - \alpha \cdot |\varphi_i|)$ are clamped to the interval $[0, 1]^2$ for the LUT.

6.3.3 *Line Integral Convolution*

Line integral convolution (LIC) was first proposed by Cabral and Leedom [31] as a texture-based vector visualization technique. The method was improved by Stalling and Hege [184]. They employed box filter kernels that reduce the total number of streamlines and make them independent from the resolution of the vector field. Instead of visualizing the vector field, the LIC is used here to illustrate surfaces. Illustrating the surface by streamlines enhances the spa-

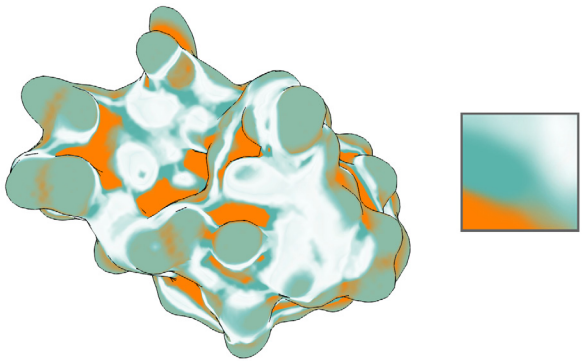


Figure 52: Illustration of the Isomerase I data set. Every vertex i is assigned to $(\tau_i, 1 - \alpha \cdot |\varphi_i|)$. The values correspond to the x- and y-component of the texture (right).

tial impression and supports the perception of shape. For interactive navigation, a frame-coherent algorithm is necessary. Therefore, the method by Huang et al. [88] is implemented, which guarantees a robust and frame-coherent LIC visualization by using customized noise textures. In the following, their algorithm will be briefly recapped. It is divided into three parts: Generating noise textures, texture projection, and creating LIC.

Generating noise textures. A sequence of mipmap textures are used to ensure a consistent LIC on the surface. The texture pyramid is build iteratively and starts with a texture I_0 of size 1×1 to a texture I_n of size $2^n \times 2^n$. Let $\eta \in [0, 0.5]$, $\Delta x, \Delta y \in \{0, 1\}$, and $\gamma_{\Delta x, \Delta y}$ be a random number uniformly distributed in $[0, 1]$. I'_n is defined as $I'_n := (1 - 2\eta)I_n + \eta$. The texture pyramid can be iteratively determined by:

$$I_{n+1}(2x + \Delta x, 2y + \Delta y) = pow\left(\gamma_{\Delta x, \Delta y}, \frac{1 - I'_n(x, y)}{I'_n(x, y)}\right), \qquad (34)$$

with $pow(n, m) := n^m$. Figure 53 shows the resulting textures.

Texture projection. First, every triangle is randomly assigned a 2D texture coordinate, which will not change during interaction. During runtime, every triangle is projected onto the noise texture regarding their assigned texture coordinate such that the map is an isometry. This ensures a consistent noise map output during the interaction. Next, the textured fragments are multiplied with the corresponding diffuse illumination to enhance the spatial impression.

Creating LIC. The illumination gradient (IG) on the surface mesh is first projected to screen space. Thus, the normalized view direction $\mathbf{v}$ and the normalized up-vector $\mathbf{u}$ of the corresponding camera are used. A 2D screen space is built with the basis $(\mathbf{u}, \mathbf{u} \times \mathbf{v})$. Afterwards, the IG $\mathbf{k}$ is projected to the screen space: $(\mathbf{Id} - \mathbf{v}\mathbf{v}^\top)\mathbf{k}$, where $\mathbf{Id}$ is

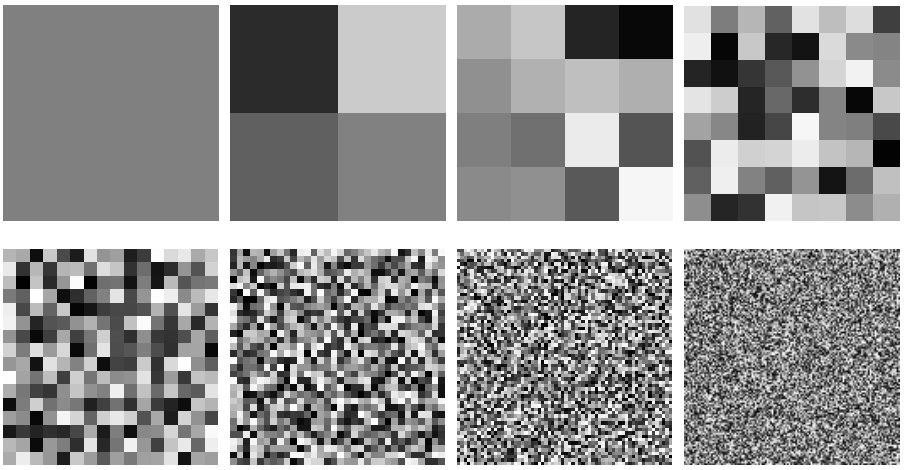


Figure 53: The first eight textures of a noise texture pyramid. The first texture has size 1×1.

the identity matrix and $\mathbf{v}$ is the normal vector of the screen space. Next, the projected IG is expressed in the basis $(\mathbf{u}, \mathbf{u} \times \mathbf{v})$, normalized, and stored in a texture (as a 2D vector). The front and back faces are used and, thus, two textures are obtained. When calling the fragment shader, the textures for LIC generation are used. For LIC propagation, an Euler method with a fixed number of iterations is used. The number of iterations is a second user-defined value. As a standard parameter, 10 iterations are suggested. Using significantly less iterations would reveal the noise textures and using more iterations would lead to blurred results. All images are taken with this setting. LICs that are close to the contour will be propagated around the surface by using the projected IG of the front faces and of the back faces.

6.3.4 *Algorithm and GPU Implementation*

The algorithm can be summarized with the scheme in Figure 54. It is entirely executed on the GPU with OpenGL shaders in a multipass rendering. First, the neighbors for every vertex are needed. For this purpose, the structure as in Section 5.3 (recall Figure 42) is employed. Based on the neighbor structure, one can determine the illumination gradient (IG) in the vertex shader, which is based on the scalar field φ. In a second render pass, the screen-based ambient occlusion (SSAO) is determined, which is based on the scalar field τ. The diffuse shading is combined with the noise textures. In the fragment shader, the triangles are projected onto the noise texture. Therefore, each fragment has an assigned portion of the noise texture. Furthermore, using the two scalar fields φ and τ, a color is assigned to the current fragment according to its $(\tau, 1 - \alpha \cdot |\varphi|)$ position in the LUT. Finally, both textures are combined via multiplication. The combined texture

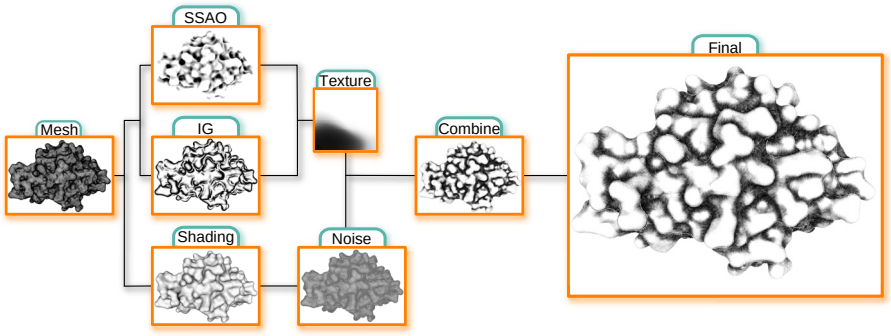


Figure 54: From a given mesh, the screen-based ambient occlusion (SSAO) term and the illumination gradient (IG) is determined. These values correspond to the RGB-values of the texture. The assigned color values are combined with the shading and the noise. Finally, LIC is applied to obtain the result.

as well as the projected IGs are stored using Frame Buffer Objects (FBO). Thus, two textures are obtained, one for the front face and one for the back face of the surface. In the last rendering pass, these textures are used to generate the LIC on the image space.

6.4 INFORMAL FEEDBACK

An informal study was performed to assess the usability of the LIC technique. The informal feedback was conducted in three steps with 12 researchers who are familiar with illustrative visualization techniques (8 men, 4 women, age of 27–37) and 3 researchers working in biochemistry (3 men, age 30–54):

1. The participants were shown different shaded models. They could explore the model and gain a 3D impression. Afterwards, different feature line methods and our feature region detection algorithm were applied to the mesh.

2. The LIC method was compared with the high-quality hatching and ConFIS. The different line drawing techniques were introduced and the participants were asked to adjust the parameters to obtain a visually pleasing result.

3. The participants were provided with different scenes and objects visualized using the LIC technique. This was used as an independent illustrative visualization technique as well as for focus-and-context visualization.

ASSESSMENT OF THE QUALITY. The goal of the first task was to assess the quality of our feature detection method, see Figure 51. The participants were asked if our method can depict salient regions. Four domain experts explicitly noted that they think that our method pro-

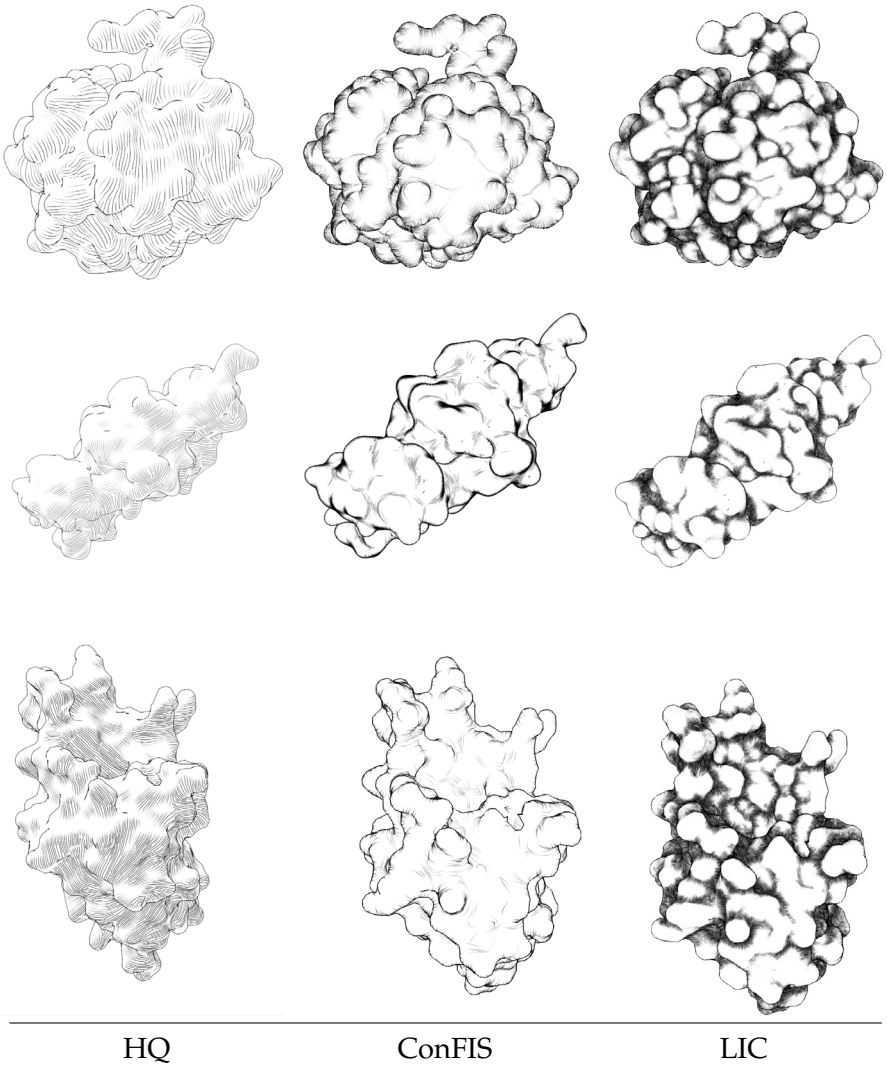


Figure 55: Examples of various models with *high-quality hatching* (HQ), *ConFIS*, and the LIC approach.

vides the same information as the feature line methods. Most participants were satisfied with the region detection, but realized that small features were detected only by conventional feature line methods unless the α parameter was adjusted to a high value. In summary, the participants stated that the feature detection method is able to detect the most prominent salient regions.

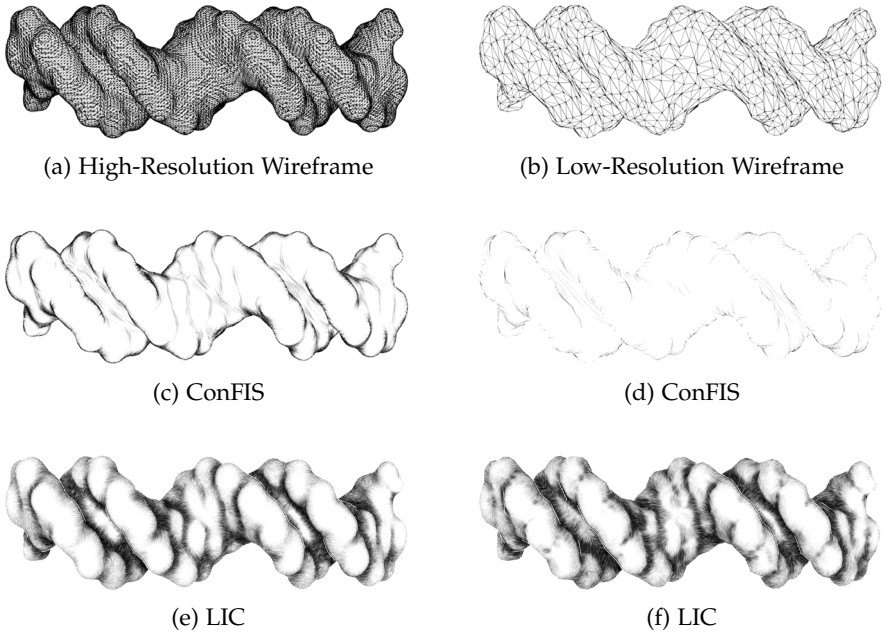
COMPARISON WITH OTHER LINE DRAWING TECHNIQUES. The second task was about a comparison with established line drawing techniques shown in Figure 55. The goal was to figure out if the new approach can keep up with the high-quality hatching [211] and ConFIS. All participants stated that all three line drawing techniques were able to give a spatial impression of the model. Most researchers were satisfied with the LIC approach. The participants were also shown a

low resolution mesh (see Figure 56) and they argued that ConFIS generates too few lines to gain a substantial impression. Furthermore, all participants liked the idea of illustrating the ambient occlusion with darker lines and therefore preferred our new method as it decodes additional information. Finally, all participants stated that the LIC approach can keep up with the compared techniques.

USE AS A FOCUS-AND-CONTEXT RENDERING TECHNIQUE. The third task was about figuring out if the LIC method can be used as an alternative for focus-and-context visualization (e.g. Figures 50b and 57b) as well as a standalone illustration. The method was not explained beforehand in task two. The users were able to intuitively figure out the meaning on their own. Here, it was noticed that the biochemists found the method more intuitive and useful than the visualization experts, since they are already acquainted with molecular data sets in general. Regarding comprehensibility, the experts first familiarized with the user-defined threshold α, which controls the feature intensity. Usually, they set it to zero and slowly increased this value until they got an impression of the surface. One visualization expert suggested combining our line drawing with toon-shading to emphasize the surface. This would also allow to color-code biochemical information (e.g. hydrophobicity or atom charge) on the surface, which one of the biochemists was missing. The focus-and-context visualization was rated as especially useful by the biochemistry experts. One of them explained that he / she would like to have the method in a molecular visualization software, since it would enhance the analyses. In contrast to classical methods that use different colors or transparency, our illustration emphasizes salient features of the surface. The participants agreed that illustrative rendering styles support the perception and reduce the distraction by unimportant parts. As the LIC method detects salient regions and illustrates them with LIC, one participant suggested to illustrate the other regions with stippling.

6.5 RESULTS & DISCUSSION

The LIC method was tested using various protein data sets from the Protein Data Bank [15] and molecular simulations provided by the project partners. The meshes were created using visual molecular dynamics (VMD) [90]. In particular, the Van der Waals (VDW) surface, the *solvent excluded surface* (MSMS [176]), and the QuickSurf molecular surface [113] were used. Van der Waals surfaces are the union of atomic spheres defined by their van der Waals radius [160]. As the *accessible surface area* describes the surface that is accessible from a solvent, the *solvent excluded surface* is the surface area of a biomolecule that excludes the accessible region [42, 167]. Therefore, it mostly consists of cavities. The experiments were conducted on a mid-range



(a) High-Resolution Wireframe (b) Low-Resolution Wireframe

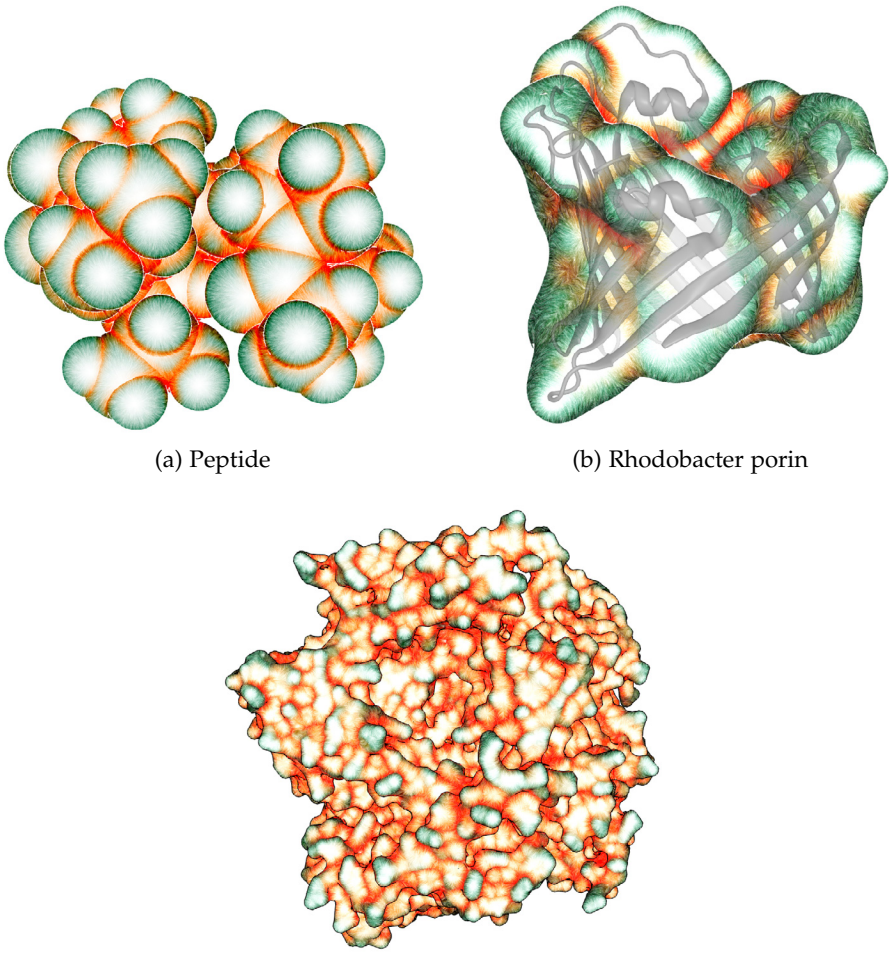
(c) ConFIS (d) ConFIS

(e) LIC (f) LIC

Figure 56: The DNA model with 65,634 ((a), (c), (e)) and 3,280 ((b), (d), (f)) triangles. The LIC method shows robust result regarding the tessellation compared with the ConFIS method. The LUT in Figure 50(b) was used.

desktop computer with an Intel Xeon CPU (3GHz), 8GB RAM, and an NVidia GeForce GTX 460. All models, numbers of triangles, and frame rates are listed in Table 6. For all models, the method can be executed in real-time. The LIC method uses two different scalar fields to encode different information. In the examples, ambient occlusion is used as a second information (τ). Furthermore, an arbitrary LUT, represented by an image, can be employed and the LIC method will illustrate the surface based on its scalar field.

The ConFIS method is able to provide illustrations that combine the advantages of feature lines and hatching methods. Therefore, the advantages of the LIC method in comparison to *ConFIS* will be listed. The first claim is that the LIC method is more tessellation-independent, see Figure 56. While the *ConFIS* method has less streamlines, the LIC method has a consistent margin where the features are depicted. This is due to the fact, that the streamlines are propagated in image-space, whereas *ConFIS* uses the barycenter of specific triangles. Hence, less streamlines are generated when the resolution is decreased. The second advantage is that the LIC method can handle deformation and depict it in real-time, see Figure 58. *ConFIS* determines the streamlines on the surface and stores them in a preprocessing step. During runtime, feature regions are determined and the streamlines are activated on these regions. Thus, a certain portion of the buffer is needed for streamline information storage. Furthermore, when changing the



(a) Peptide

(b) Rhodobacter porin

(c) Potassium Channel

Figure 57: Different molecules illustrated with the LIC approach: (a) shows a small peptide as VDW surface; (b) is a focus-and-context rendering of a porin, where a smooth molecular surface and a secondary structure (cartoon) representation are combined; (c) illustrates the solvent excluded surface of a potassium channel. The LUT shown in Figure 52 was used for all images.

underlying vector field, e.g., due to mesh deformation, the *ConFIS* method has to recompute the streamlines, which is time-consuming. For the LIC method, not the same amount of buffers are needed, as the LIC is illustrated in real-time. Thus, the LIC method is more tessellation-independent, yields reasonable results, and can handle deformations in real-time, see Table 6. As mentioned in Section 6.2, illustrative rendering has been shown to effectively support the visual analysis of molecular surface data. In most cases, however, only simple methods like contour lines or ambient occlusion are used. The goal was to illustrate complex models like molecular surfaces, which is important to analyze the protein function. More specifically, the shape of the molecular surface is illustrated and salient features are

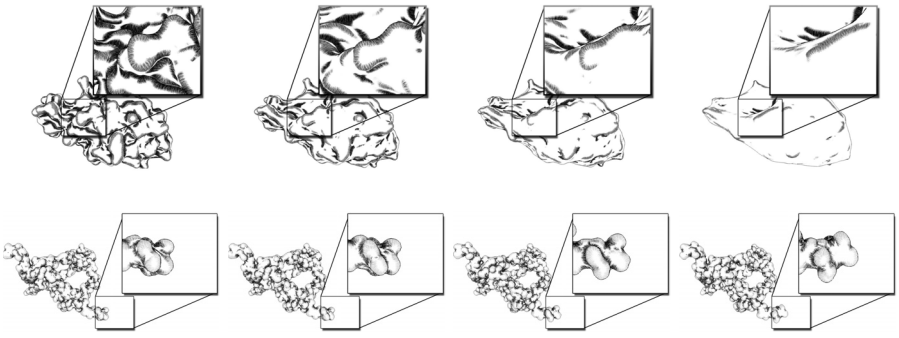


Figure 58: The Isomerase II (upper row) and hydrogel model in different animation steps with the LUT in 50(b). The upper row shows a gradual change of the surface parameter. The lower row shows different time steps from a simulation trajectory.

highlighted, like indentations and gaps, in the outer surface (so-called cavities or pockets). These features are often an indicator for binding sites, which are crucial to protein functions like enzymatic reactions.

As the surface visualization uses not only the illumination gradient, but also a second parameter, it can bring out different criteria. For molecular surfaces, ambient occlusion serves well, since it provides evidence of indented surface parts. Using an appropriate LUT, the two parameters can be either illustrated individually (cf. Figure 52) or combined (cf. Figure 50a). Besides illustrating the shape of the surface, a second intended use case for the LIC method was focus-and-context visualization.

Often, users want to see a combination of two molecular models when analyzing protein data, because they highlight different aspects of the protein. For example, the molecular surface shows the interface to other molecules while the cartoon representation shows the functional (secondary) structure of the protein. Usually, the surface is rendered semi-transparently around the other model. Since the molecular surface is generally very uneven, this results in a lot of visual clutter, which makes it hard to identify interesting features in both of the models. Figure 57b shows a combined visualization where the illustrative LIC method was applied to the molecular surface in order to provide the context for the internal cartoon representation. Similar methods have been used successfully in medical visualization [195]. Focus-and-context is also essential for visualizing molecular complexes that consist of several identical subunits. Subunits that are in focus can be visualized using classical shading while the others are rendered using the LIC method in order to provide context as shown in Figure 50b. As described in Section 6.4, the initial feedback of the project partners from biochemistry was overall very positive. They liked the visual appearance of the LIC method and believed that it would be beneficial for visual data analysis.

Table 6: Rendering performance of the LIC approach in frames per second (*Our*). We also provide the frame rate of pure Phong shading as a reference (*Phong*).

Model	#Δ	**FPS**	
		Phong	LIC
Rhodobacter porin (Fig. 57b)	30,732	191	120
Peptide (Fig. 60a)	51,200	137	78
Left ventricle (Fig. 59b)	61,192	145	81
DNA (Fig. 56)	65,634	131	75
Hydrogel (lower row, Fig. 58)	71,702	130	73
Isomerase II (1^{st} step, Fig. 58)	78,672	127	61
Isomerase I (Fig. 50a)	83,616	110	73
Aneurysm (Fig. 59a)	98,970	101	61
Lipase (Fig. 55c)	122,867	83	23
Potassium Channel (Fig. 57c)	207,429	52	19
ATPase (Fig. 50b)	558,640	79	15

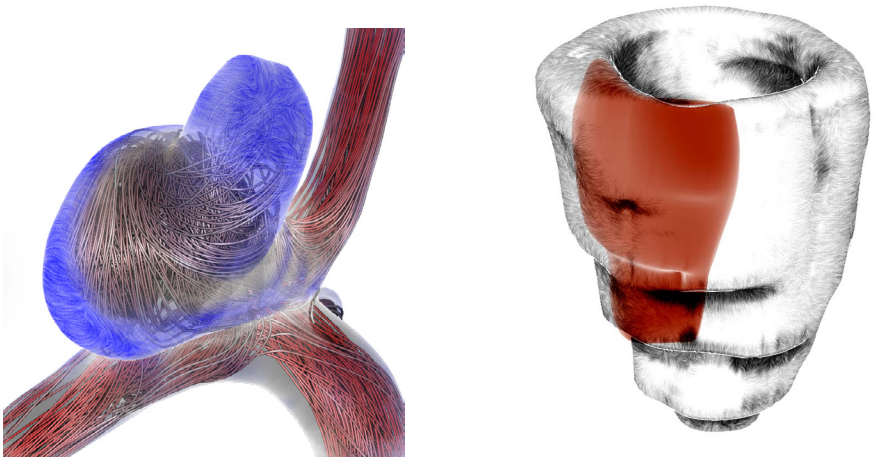
6.6 FURTHER APPLICATIONS

Although the LIC method was designed for the illustration of molecular data, it is a general technique for illustrative rendering. In this section, its applicability to medical visualization as well as for artificial data sets is shown.

Exploration of blood flow. In case of cerebral aneurysms – a special kind of vascular disease – the medical researcher is interested in the qualitative exploration of near-wall hemodynamics. Neugebauer et al. [229] presented a framework for this investigation. They detected regions of interest on an aneurysm using the *shape index* to examine the underlying blood flow. Afterwards, they visualized the underlying streamlines on a 2D widget. The shape index is proposed as the scalar field $y = \varphi$ and the vector field on the surface is used as the projection of the underlying blood flow data. Thus, φ is set as the shape index:

$$\varphi = \frac{1}{2} - \frac{1}{\pi} \arctan \frac{\kappa_1 + \kappa_2}{\kappa_1 - \kappa_2} \tag{35}$$

with $\kappa_1 \geqslant \kappa_2$ as the principal curvatures. Here, φ ranges from 0 (cup) to 1 (cap). Thus, images are obtained which can be used for the qualitative exploration. The streamlines on the surface are obtained. Therefore, the LIC method can convey both information (shape index and streamlines) in a single 3D scene.

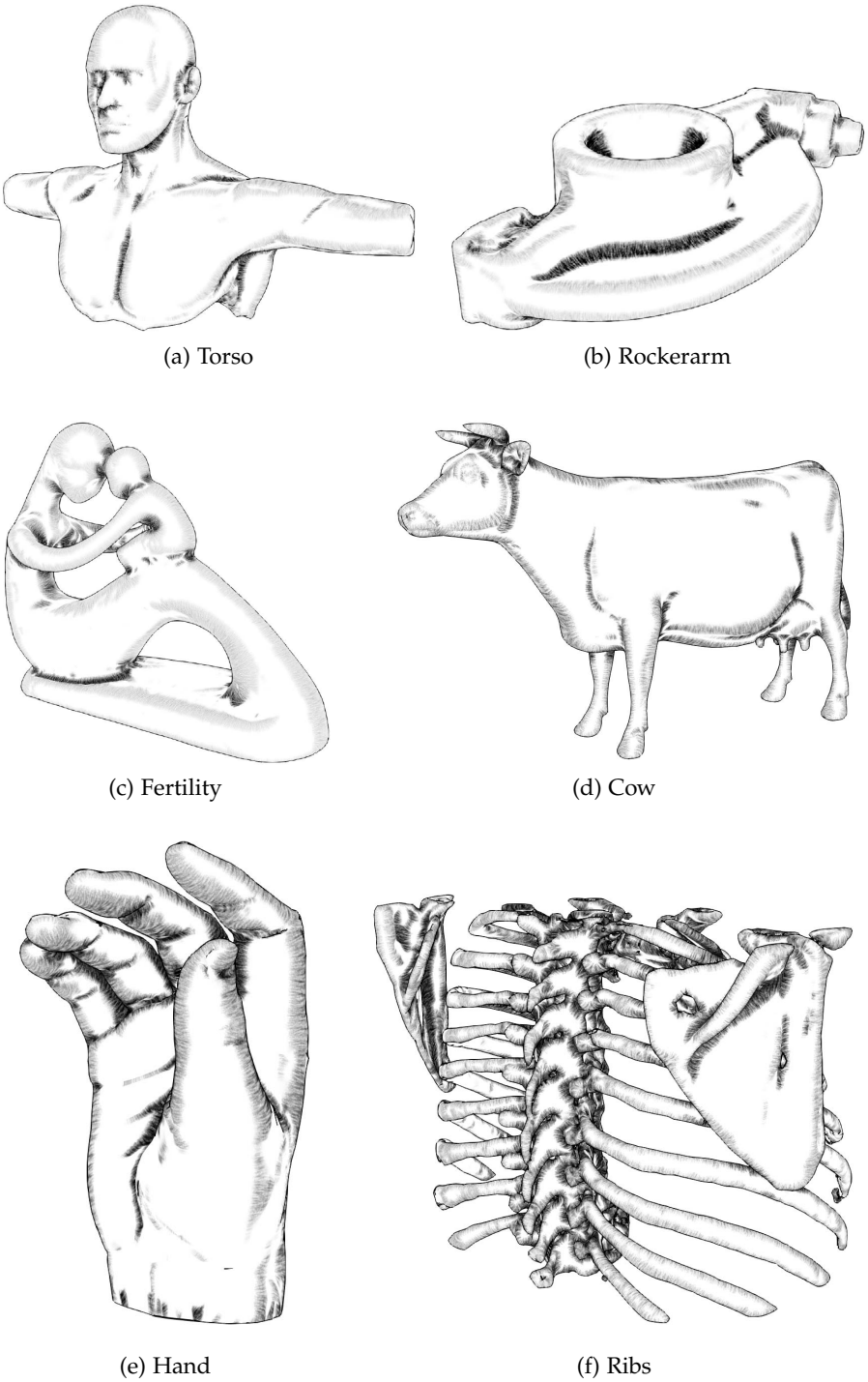


(a) Cerebral aneurysm (b) Left ventricle

Figure 59: (a) Qualitative exploration of the hemodynamics (aneurysm); (b) Focus-and-context visualization with the left ventricle and their interior infarction scar depicted in orange.

Myocardial functional data. In the field of focus-and-context visualization, the LIC method is used to illustrate a surrounding object in order to distinguish it from the interior objects. The LIC technique is employed to the left ventricle and the infarction scar, resulting from a heart attack, is shaded in red. Oeltze et al. [154] used a glyph-based approach to encode the thickness of the left ventricle. Additionally, φ can be encoded as the wall thickening of the ventricle. The clinical experts evaluate the heart muscles wall thickening to assess the severity of a heart attack. The LIC technique is appropriate for this application, since it is able to depict the contraction of the ventricle. Therefore, the domain expert is able to observe the left ventricle during the deformation and can still gain information from wall thickening from the LIC method. Figure 59 shows two examples for the discussed application areas where the LIC method can be used for focus-and-context visualization.

Artificial Datasets. As stated earlier, the LIC method was motivated for molecular surface rendering, but not restricted to it. Here, the LIC method is applied to artificial models to emphasize that this method is also applicable to deliver pleasant results. Therefore, six artificial models are chosen and depicted in Figure 60: torso, rockerarm, fertility, cow, hand, and ribs. Here, the scalar fields φ and τ are used as described in Section 6.3.2. Furthermore, the 2D LUT as shown in Figure 50b is used for all examples.



(a) Torso (b) Rockerarm

(c) Fertility (d) Cow

(e) Hand (f) Ribs

Figure 60: Various surface models illustrated with the LIC method. Besides the molecular surfaces, the presented method is also able to give visual pleasant results.

6.7 CONCLUSION

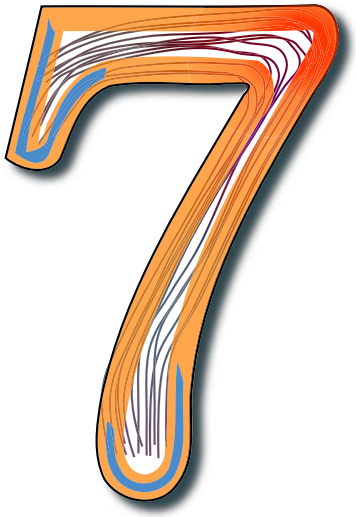
A novel illustrative visualization technique was presented for molecular structures. The method detects salient regions and illustrates them

using LIC. Hence, the user gets a spatial impression of the surface. The LIC is used to gain an enhanced impression of the curvy surface. Salient region detection is based on the illumination gradient. Furthermore, 2D LUT is used to encode regions of interest as well as to color-code different properties on the surface. Therefore, two scalar fields on the surface are proposed and their values are used to assign the color of the LUT in their corresponding coordinates. One advantage of the described method is that it can handle animated surfaces and illustrate them in real-time. This work is motivated by molecular visualization but not restricted to it. Furthermore, the LIC approach provides a frame-coherent illustration and is more tessellation-independent in comparison to *ConFIS*.

Part III

VESSEL VISUALIZATION

Adaptive Vessel Visualization



This section is partly based on:

Kai Lawonn, Rocco Gasteiger and
Bernhard Preim
Adaptive Surface Visualization of Vessels
with Embedded Blood Flow Based on the
Suggestive Contour Measure
Vision, Modeling and Visualization, pp. 113–120, 2013

Kai Lawonn, Rocco Gasteiger and
Bernhard Preim
Adaptive Surface Visualization of Vessels
with Animated Blood Flow
Computer Graphics Forum (in print), 2014

7 ADAPTIVE VESSEL VISUALIZATION

7.1 INTRODUCTION

The initiation and evolution of cardiovascular diseases (CVDs), such as cerebral and abdominal aneurysms, are multifactorial problems involving hemodynamics, wall biomechanics, genetics, vessel morphology, and other, not well understood, factors [3]. In recent studies, domain experts identified certain hemodynamic information as important indicators for the presence, initiation, and outcome of a CVD [35, 134]. The hemodynamic information comprise quantitative measures (e.g., wall shear stress (WSS), pressure, speed) and qualitative characteristics (e.g., inflow jet, degree of vorticity), which describe the blood flow behavior. These information are derived from patient-specific flow measuring with time-resolved phase-contrast magnetic resonance imaging (4D PC-MRI) [134] or computational fluid dynamics (CFD) simulations [34, 104].

Furthermore, for particular CVDs, such as cerebral aneurysms, CFD simulations provide clinically relevant information regarding treatment options by conducting virtual treatments [3]. The acquired hemodynamic information are very complex because they consist of several multivariate (e.g., scalar and vector data) and multidimensional (3D and 4D) information. In addition to a quantitative analysis, an effective visual exploration of these attributes is important to obtain insights based on this information. For the visual exploration, domain experts, such as biomedical researchers and CFD blood flow experts, are interested in the hemodynamics *and* the surrounding vessel anatomy because both information are strongly correlated to each other [7]. On the one hand, this leads to an embedded surface problem where the flow is occluded by the surrounding vessel surface. These occlusions have to be resolved in such a way that important anatomical surface features are still depicted, whilst simultaneous visibility of the embedded flow visualization is ensured. On the other hand, the dynamic behavior of the time-dependent flow datasets has to be conveyed, especially the correlation between near-wall flow patterns and surface features. It is assumed that abnormal near-wall flow leads to pathological vessel dilatations and may increase the risk of a new rupture at locations where a former rupture occured.

Existing techniques for displaying embedded objects within enclosing objects, such as semitransparent rendering or clipping of the vessel surface, exhibit a reduced surface shape depiction and ambiguities of the spatial relationship between vessel sections. This decreases the

observers' ability to mentally link the vessel morphology with the internal flow visualization. Gasteiger et al. [66] tackled this embedded surface problem and proposed an adaptive surface visualization that incorporates a ghosted view approach. The ghosted view leads to a completely occluded flow visualization below surface regions that are facing away from the viewer. Furthermore, the view-dependent opacity does not ensure an appropriate depiction of salient concave and convex surface regions, which are necessary to identify pathological bulge formations of the vessel wall.

Inspired by illustrative line rendering techniques, an adaptive surface rendering is proposed that overcomes these limitations. Therefore, a surface shading was developed that incorporates the view-dependent curvature of suggestive contours into the ghosted view approach of Gasteiger et al. [66]. This ensures both visibility of the embedded flow visualization and expressive depiction of salient vessel surface features as well as highlighting nearby surface regions from animated pathlines.

In summary, the contributions of this chapter are:

- A novel technique for vessel visualization with embedded blood flow information depicted with established flow visualization methods.
- A method which was adapted from an established feature line technique - *suggestive contours*.
- A technique, which is applicable to arbitrary vessel surfaces, but also to other patient-specific anatomy.
- An animated pathline approach where nearby surface regions are highlighted to gain spatial information of the interior blood flow.

7.2 MEDICAL BACKGROUND AND REQUIREMENT ANALYSIS

CVDs refer to the class of diseases that affect the heart and blood flow vessels (arteries and venes). Common examples of CVDs are aortic aneurysms and dissections, cerebral aneurysms, or atherosclerosis. For the identification, progression, and risk assessment of CVDs, the blood flow behavior plays an important role. A particular example are cerebral aneurysms, which are pathological dilations of the vessel wall that exhibit an increased risk of rupture [101]. Risk factors and other relevant flow characteristics are identified by quantitative and qualitative analyses. This chapter focuses on the qualitative analysis, which involves a visual exploration of the morphology *and* its embedded blood flow information. Domain experts require the visualization of both information because they strongly influence each other. For example, in the case of cerebral aneurysms, a bleb formation indicates

a previous rupture and increases the risk of further ruptures. Blebs are local bulges on the aneurysm sac and can be found at regions of high WSS and near the flow impingement zone. Thus, an expressive surface depiction that conveys such morphological features, but ensures both visibility of the underlying flow and indication near-wall flow is necessary.

7.2.1 *Requirement Analysis*

A detailed visual description of the enclosing vessel surface leads to an increased occlusion of internal information. Thus, an adapted surface visualization is needed to reduce the occlusion. Based on literature [33, 131] and discussions with domain experts, the following requirements are addressed:

Visibility of internal flow information: A maximum visibility of the internal flow visualization is required during the visual exploration. With "maximum visibility", as few as possible occlusions of the flow visualization are meant by the enclosing surface. This supports the viewer in interpreting and tracing the flow.

Emphasis of relevant surface features: As vessel morphology and flow influence each other, an expressive vessel shape depiction is necessary that conveys surface features such as concave and convex regions as well as bleb formations.

Revealing pathline-nearby surface information: Since hemodynamic information, e.g., speed, pressure, and vorticity, is important for blood flow assessment, an animated pathline animation has to be provided. To improve the perception of the position of the animated pathlines, a highlighting of nearby surface regions is favorable.

Improvement of depth perception: For the depiction of overlapping and distant vessel parts, depth cues should be provided. These hints improve the perception of depth and spatial relationships of the vessel surface and attract the attention to vessel regions that are close to the viewer.

7.3 RELATED WORK

Effective embedded surface visualizations are relevant in several domains like engineering, vector field analysis, medical research, and treatment planning. Each scenario is faced with occlusions and challenges regarding perception of shape, depth, and spatial relationship. Thus, several visualization domains are involved to cope with these challenges.

VISUALIZATION OF EMBEDDED STRUCTURES. Interrante et al. [92] investigated how sparsely-distributed opaque texturing can be used to depict the shape of transparent iso-intensity surfaces of radiation dose as a special instance of an embedded surface visualization problem. An interactive view-dependent transparency model was proposed by Diepstraten et al. [48] to improve the shape perception of embedded structures. Based on several *design rules*, the transparency and visibility of the layered objects are adjusted according to the camera view and the spatial relationship between opaque and semi-transparent objects. This kind of visualization is an example of ghosted views and belongs to the group of *smart visibility techniques*. These techniques focus on exposing the most important visual information and originate from technical illustration. Other examples are cut-away views, section views, and exploded views [202]. From an applicative view, these methods can also be used for intervention planning and disease understanding. Ishida et al. [95] applied stream and particle tracing for visual exploration of measured blood flow in cerebral aneurysms.

A multipass framework for illustrative rendering of complex and self-occluded integral surfaces was proposed by Hummel et al. [89] and Born et al. [19]. They incorporated several rendering techniques, such as transparency modulations, hatching textures, halftoning, and illustrative streamlines, to reveal subjacent layers and to enhance shape and depth perception of each layer. The visualization of the intrinsic blood flow can be further combined with line predicates that enable selective pathline bundles, e.g., to detect regions with high vorticity [18]. In Gasteiger et al. [66], a multipass framework was presented that incorporates some of the design rules of Diepstraten et al. [48] to achieve a ghosted view for enclosing vessel surfaces with embedded flow information. The opacity of the vessel surface is controlled by a Fresnel opacity term, which adaptively changes the transparency dependening on the view direction. Baer et al. [6] confirmed the performance of the visualization in a quantitative user study. This study is strongly related to the studies by Koendrink et al. [109], Kim et al. [106], and Blair and House [8]. However, a limitation of the ghosted view method becomes obvious in regions that are oriented away from the viewer and occlude the underlying flow visualization. Additionally, salient surface regions described by concave and convex regions may not be well conveyed by the Fresnel opacity.

ILLUSTRATIVE FLOW VISUALIZATION. For an extensive overview of flow visualization, the state-of-the-art surveys by McLoughlin et al. [139] (geometric-based), Salzbrunn et al. [175] (partition-based) and Laramee et al. [122] (topology-based) are recommended. Traditional visualizations of time-dependent vector fields can be found in [120, 121]. In Mattausch et al. [137] and Everts et al. [58] illustra-

tive line style methods, such as halos, glyphs and depth attenuations, were utilized to enhance the depth perception of dense streamline and pathline datasets. Van Pelt [159] presented real-time illustrative visualization and exploration methods for measured cardiac blood flow. Markl et al. [131] presented methods for the acquisition and illustration of the blood flow in heart and large vessels. Born et al. [20] and Köhler et al. [110] introduced approaches for the extraction and illustrative depiction of flow structures in measured cardiac blood flow based on the concept of line predicates. This is motivated by research findings in cardiology that confirm correlations between some cardiac diseases and certain flow patterns. Gasteiger et al. [67] introduced the *FlowLens*, an interactive focus-and-context tool for the exploration of multiple flow attributes for simulated and measured blood flow. For example, the animation of pathlines could be observed within the lens and relevant context attributes, such as WSS and pressure, outside the lens. A system for the qualitative exploration of near-wall hemodynamics in cerebral aneurysms was presented by Neugebauer et al. [229]. Hope et al. [86] provided an overview of aortic imaging and visualized the aortic blood flow. A recent overview about visual exploration methods and future visualization challenges for simulated and measured flow datasets are given by Preim and Botha [164] and Vilanova et al. [201], respectively.

DEPTH AND SPATIAL RELATIONSHIP. The application is also related to perception-based 3D graphics. Two important cues for depth and spatial arrangement are shadowing and shading [204]. A survey about existing shadow rendering approaches can be found in Liu and Pang [129]. In Luft et al. [130], an image-based method was presented to efficiently integrate depth cues into complex scenes based on the differences between the depth buffer and its low-pass filtered copy. Further depth cues are atmospheric attenuation, depth blurring, and line fading, which are discussed in Svakhine et al. [188]. The shadow approximation proposed by Luft et al. [130] and atmospheric attenuation were also utilized by Gasteiger et al. [66] to enhance the depth perception of vessel regions. Shadow-like depth indicators by means of an adaptive hatching method were proposed by Ritter et al. [168] to support reliable comparisons of spatial distances in complex vascular structures. A weighted combination of illustrative rendering techniques was utilized by Tietjen et al. [196] for depth- and shape-enhanced medical surface visualizations inspired by medical textbook illustrations. The weighting is controlled by a *shading map*, which combines several illumination and surface information such as plateau and raking light, atmospheric attenuation, and curvature.

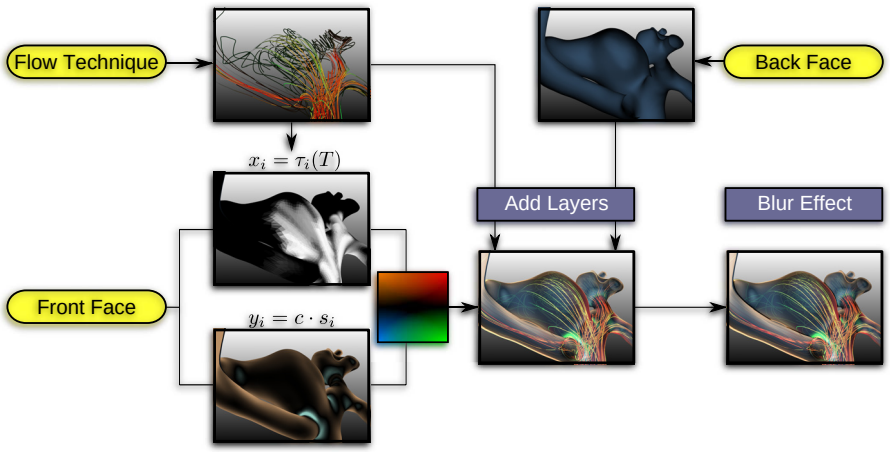


Figure 61: Overview of the novel adaptive surface visualization: First, the blood flow with established flow visualization techniques, e.g., pathlines is visualized. Second, the shading of the front faces is determined. This calculation is based on the (x, y) coordinates. For the x coordinate, distances of the pathlines to the surface are used. The y coordinate is determined according to the suggestive contour measure. The back faces are rendered opaque. Finally, each visualization layer is composed and shadow approximation as well as depth blurring are added to the result.

7.4 METHOD

The presented method for the novel adaptive surface visualization is based on the multipass framework proposed by Gasteiger et al. [66] and incorporates different visualization techniques, as illustrated in Figure 61. The ghosted view approach is improved by modifying the vessel opacity of the front faces by means of the suggestive contour measure, see Section 7.4.1. Furthermore, depth blurring is utilized instead of atmospheric attenuation because of the more natural depth perception. First, a previously mentioned feature detection and its incorporation into the modified ghosted view shading approach will be presented.

7.4.1 *Feature Regions*

According to Section 6.3, a scalar field on the surface is determined. For the detection of a feature regions, a scalar field is employed which was properly used for the LIC method. Therefore, the scalar field φ, see Equation 32, is used:

$$\varphi_i = D_{w_i} l_i = \langle \nabla l_i, w_i \rangle. \tag{36}$$

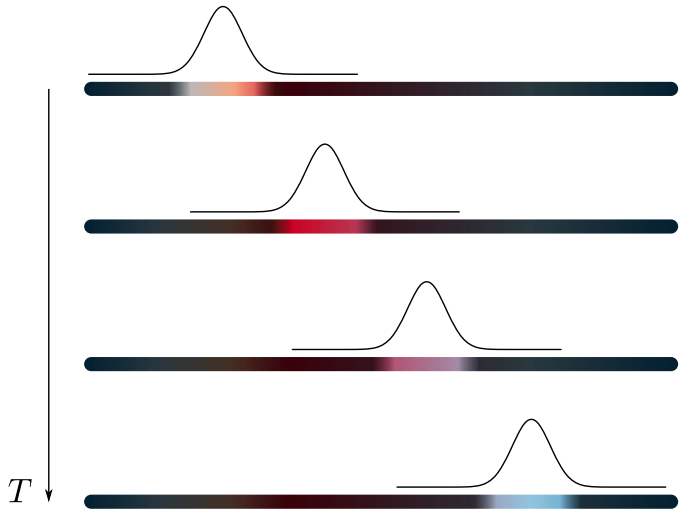


Figure 62: A pathline with additional color-coded information is illustrated. During the animation a Gaussian distribution runs over the pathline to highlight corresponding parts.

7.4.2 *Pathline Animation*

Domain experts are also interested in the dynamic behavior of the time-dependent flow characteristics. Therefore, pathlines were extracted out of the flow datasets with an adaptive Runge-Kutta scheme [194]. To support the blood flow assessment, the pathlines are animated over time to convey speed information. However, the whole pathline is visualized and color-coded with additional information, such as curvature or vorticity, as context information. For avoiding visual clutter, desaturated colors are used and parts of the pathlines are enhanced with a Gaussian distribution during the animation, see Figure 62. In addition, a scalar field $t(\mathbf{q})$ as time information is given for each individual pathline. This scalar field represents the pathline point $\mathbf{q}$ in time where the pathline passes the corresponding surface location. For example, having the time $t(\mathbf{q})$ at point $\mathbf{q}$ means that the pathline passes $\mathbf{q}$ at that time $t(\mathbf{q})$. In detail, the distribution for the animation of a single pathline is used:

$$f(T, \mathbf{q}) = \alpha \cdot e^{-\frac{(T-t(\mathbf{q}))^2}{\beta}},$$

where T denotes the animation time. Therefore, each pathline PL has a maximal time length $t'(PL)$. As the points of the pathline $\mathbf{q}_i \in PL$ with $i \in \{0, \ldots, N\}$ are consecutively ordered it yields $t'(PL) = t(\mathbf{q}_N)$. For the animation, the value T runs from 0 to the maximal time value $\max_{PL} t'(PL)$. The variable α is set to $\alpha = 1$ because this illustrates the original color of the pathline at $\mathbf{q}$ with $T = t(\mathbf{q})$. The parameter β represents the length of highlighted parts on the pathline. The smaller β, the smaller the highlighted part. The variable β is determined by computing the median (med) of the animation times $t'(PL)$ of all

pathlines: $\text{med}_{PL}\, t'(PL)$. Finally, β is set to $\beta = 2 \cdot \text{med}_{PL}\, t'(PL)$ as a default value, but the user can adjust it during runtime.

The highlighting function f is multiplied with the color coding of the pathlines. To avoid rather dark areas, f is clamped to the interval $[0.25, 1]$. Therefore, the pathlines are still perceivable even if the animation at the current time point does not result in a highlighting of a pathline. Furthermore, the highlighting function f is chosen as to be time-dependent regarding the integration time. Thus, having a pathline that runs faster through the vessel than the other, the highlighted parts are wider. This supports the perception of fast pathlines and is based on the idea of a trail that is longer when the object is faster.

7.4.3 *Highlighted Surface Regions*

To improve the visual exploration of the animated pathlines, an enhanced illustration of the pathline-nearby surface region is employed. Therefore, a time-dependent scalar field $\varphi(\mathbf{p}, T)$ on the surface mesh is determined. This scalar field contains the minimal distance of all highlighted pathline parts regarding the animation time T. Thus, the scalar field $\varphi(\mathbf{p}, T)$ encodes the minimal distance from surface point $\mathbf{p}$ at time T to a highlighted pathline part.

Next, this scalar field is transformed in such a way that surface parts distant to highlighted pathline parts are assigned to zero, whereas nearby regions are at most 1. This choice will be explained in Subsection 7.4.4. The scalar field is transformed by:

$$\tau(\mathbf{p}, T) = 1 - \tfrac{1}{\gamma} \cdot \varphi(\mathbf{p}, T).$$

For the surface mesh, the position of the vertex is indicated by $\tau_i(T) \coloneqq \tau(\mathbf{p}_i, T)$. The parameter γ can be seen as a threshold for distances. The variable γ is determined by calculating the 1-percent quantile, namely percentile. Therefore, γ is set as the 1-percent quantile of the set of distances ordered from low to high. This ensures that τ is higher than zero for values less than the 1-percent quantile and less than zero for values higher than the 1-percent quantile.

7.4.4 *Color Assignment*

The shading is defined by two values and a lookup table (LUT), see Figure 63. The first value per vertex is given by

$$y_i = c \cdot s_i = c \cdot (\mathbf{w}_i \cdot \nabla l_i)$$

with the values defined in Subsection 7.4.1. Here, c is a user-defined value, which adjusts the brightness of the shading. Prominent highlights are defined as zero-crossings of s_i and these regions are conveyed with an expressive color coding. Thereby, negative and positive

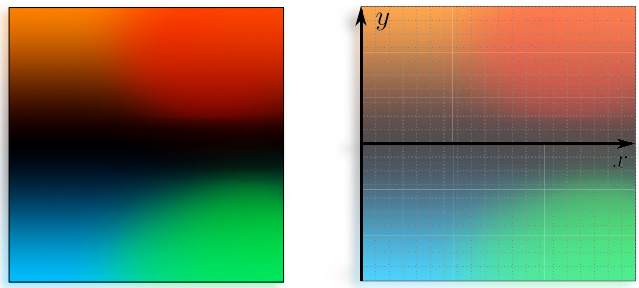


Figure 63: The 2D texture lookup table LUT and its coordinate system that represents the color coding for concave (cyan) and convex (orange) as well as pathline-near (red) and pathline-far (green) vessel surface regions.

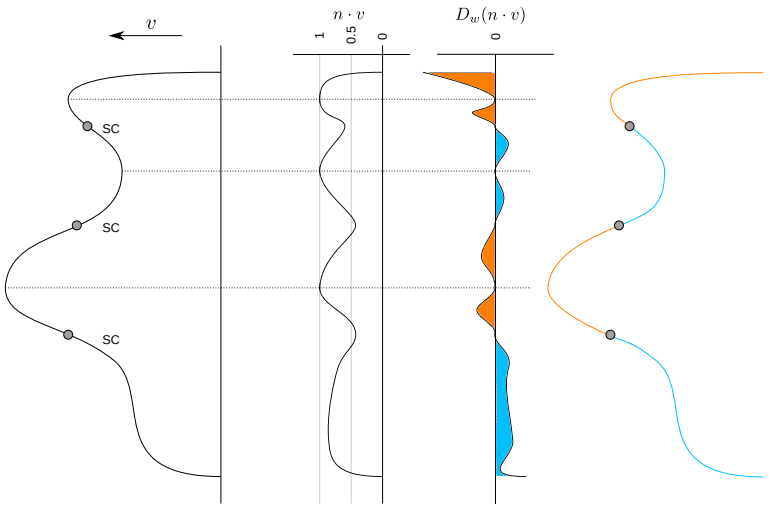


Figure 64: The surface mesh in the profile with suggestive contours denoted as SC. From left to right, the diffuse light values $\langle \mathbf{n}, \mathbf{v} \rangle$ and the values of Equation 32 $D_{w_i} l_i$ are illustrated in orange and cyan. The segments with corresponding positive or negative values are depicted.

values of s_i are differentiated, see Figure 64 for an illustration. The second value per vertex is given by

$$x_i = \tau_i(T) = 1 - \frac{1}{\gamma} \cdot \varphi(\mathbf{p}_i, T)$$

with the values defined in Subsection 7.4.3. The assignment of the colors for each vertex is performed by a 2D lookup table (LUT). The values (x_i, y_i) are used to assign the defined color to vertex i. This is defined by the RGB value position (x_i, y_i) in the LUT. The LUT has coordinates $[0, 1] \times [-1, 1]$ and therefore (x_i, y_i) is clamped to $[0, 1] \times [-1, 1]$ as well. Figure 65 shows an example of the color LUT application during the animation.

To differentiate the regions, two antipodal colors in the CIELab colorspace are chosen. First, a color $col_1 = \{L, a, b\}$ is chosen, in this case orange with $col_1 = \{65, 51, 74\}$, and the signs of a, b are changed.

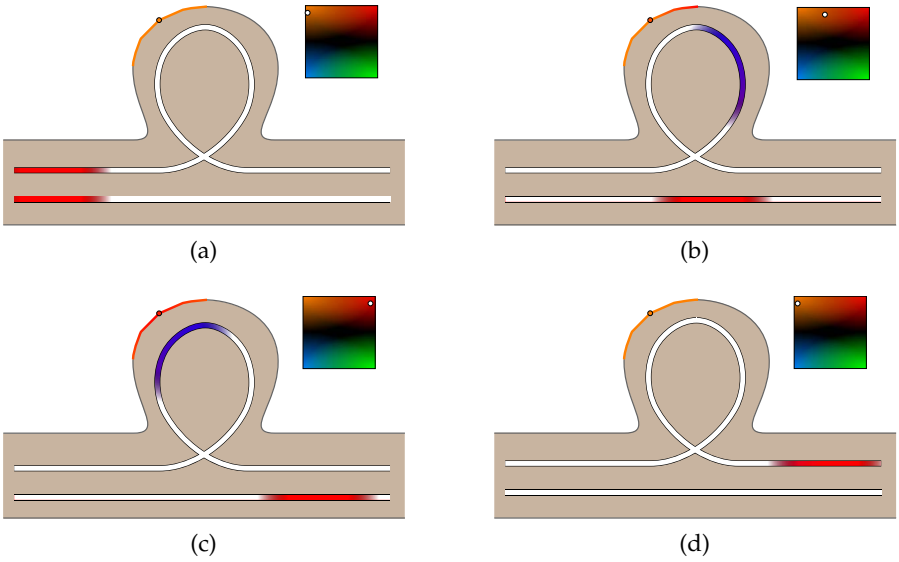


Figure 65: A schematic view of the proposed approach. First, every point is assigned to (x, y) coordinates of the LUT. In (a) to (c) the x coordinate changes because the highlighted pathline gets closer to the point on the surface. In (c), the pathline is closest to the point and therefore obtains the color red. Note that in this example the viewpoint is always the same and therefore no change of the y coordinate occurs.

Thus, $col_2 = \{L, -a, -b\}$ is obtained, in this case $col_2 = \{65, -51, -74\}$, cyan. Thus, two different colored regions are assigned where the border represents a feature derived by *suggestive contours*. These colors are used for the y-axis of the LUT. Orange represents the positive y values and cyan the negative ones. Both colors have a linear color gradient from black at $y = 0$ to $y = 1$ orange and accordingly to $y = -1$ cyan. For the x-axis, the H value from the HLS color space is decreased such that red for $(1, 1)$ and green for $(1, -1)$ is obtained. In summary, a LUT with $(x, 0) \triangleq$ black, $(0, 1) \triangleq$ orange, $(0, -1) \triangleq$ cyan, $(1, 1) \triangleq$ red, and $(1, -1) \triangleq$ green is assessed. The transitions of the colors are linear. Additionally, the color on the surface mesh is determined and it is composed with the backface surface color. The result is a shading, which conveys the impression of a Fresnel opacity with highlighted surface features.

7.4.5 *Visual Effects*

Some visual effects are added to emphasize the region of interest. First, an approximated shadow casting onto the front faces of the vessel surface is applied to enhance the spatial relationship between overlapping vessel sections. Here, the method proposed by Luft et al.[130] was implemented and a *spatial importance function* from the depth buffer and its low-pass filtered version was computed. The

spatial importance function is determined by applying a Gaussian filter kernel to the depth buffer and subtract it with the depth buffer. The low-pass filtering is accomplished by the *spatial importance function*. Negative values represent areas of background objects that are close to other occluding objects. The shadow casting is approximated by adding the negative values to the original color values, which causes a local darkening.

Furthermore, a focus region is considered as basis for focus-and-context visualization as well as depth attenuation. This region is defined by a depth-near and depth-far region adjusted by the user. Thus, pathological regions, such as the aneurysm sac or a stenosis, can be emphasized. Vessel sections, which are outside this region, will be blurred according to a Gaussian blurring filter with a kernel size that linearly depends on the distance between vessel sections and the focus region. This leads to a smooth transition between the focus region and the surrounding that supports both attraction to the focus region and perception of depth. The blurred vessel sections with their embedded flow visualization are still provided as context information (see Fig. 61, right).

7.5 ALGORITHM AND IMPLEMENTATION

The algorithm of the shading comprises the following steps:

1. (Optionally) Subdivide and smooth the mesh.
2. Determine distances of the surface to the pathlines.
3. Compute vertex normals.
4. Build neighbor information.
5. Determine vertex gradients.
6. Compute color for each vertex.
7. Blur the rendered image and add shadows.

The algorithm is divided in two parts. The first part (1-4) consists of several preprocessing steps and the second part (5-7) is the rendering loop. Both are described in more detail in the following sections.

7.5.1 *Preprocessing*

At the beginning, the time-dependent scalar field on the surface mesh is determined that encodes the distances of the pathlines. For every vertex, the distances to the pathline points are measured and the shortest one for each time step is chosen. Therefore, several scalar values for each vertex are obtained, which encode the distances for

every time step. The values are stored by using a rectangle texture GL_TEXTURE_BUFFER. This texture has size $\#vertices \times \#timesteps$ and stores the values of x_i, as described in Section 7.4.3. The determination of the minimal distances is computationally intensive. As this computation is a preprocessing step, the distances for each vertex and each point on the pathlines are compared. An improvement of this method would be to use k-d trees [14] and adapt them to the GPU [214].

7.5.2 *Rendering Loop*

During runtime, steps 5, 6, and 7 are executed in a two-pass rendering. In the first pass, the vertex gradient is determined in the vertex shader, which depends on the camera position and the light position. Subsequently, the pixel colors are computed and assigned in the fragment shader as described in Section 7.4.3 and Section 7.4.4. In the second pass, the Gaussian blurring and shadowing is applied as presented in Section 7.4.5.

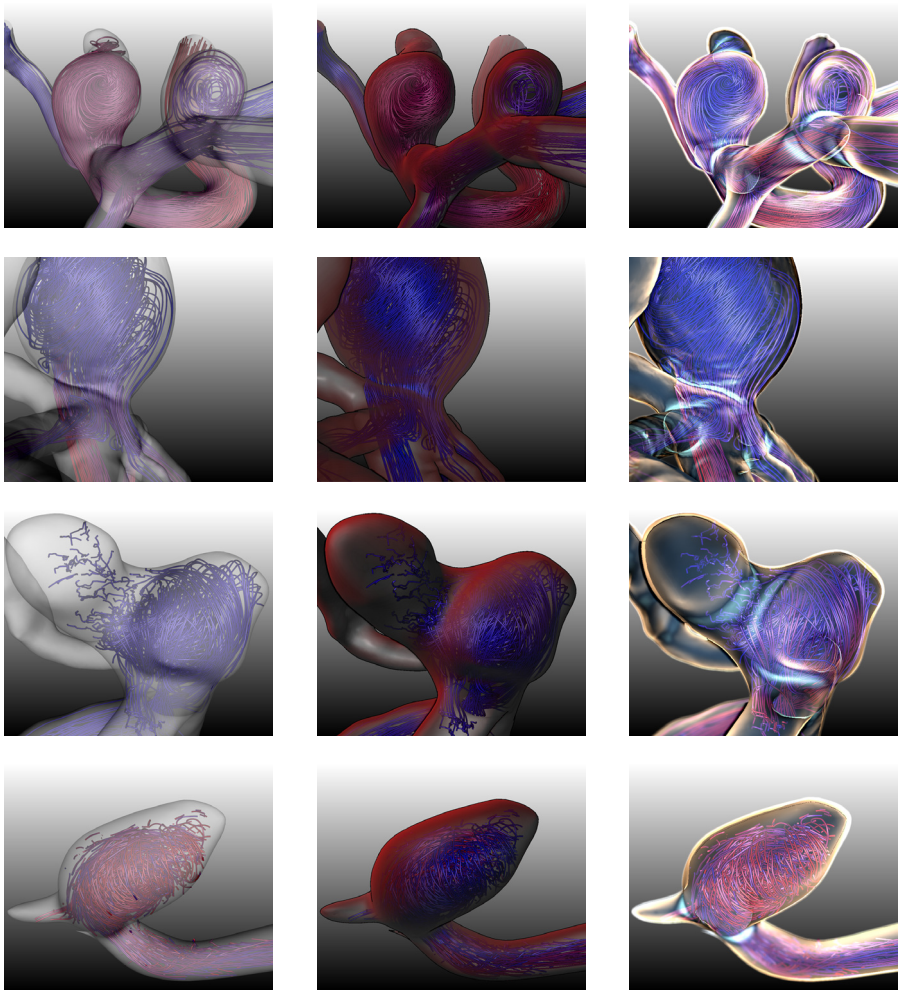
During the pathline animation, the user is able to adjust typical animation settings such as: play, stop, next frame, previous frame, slower, and faster. The animation stops if the user rotates the scene to avoid visual distractions during the exploration. Therefore, the user can adjust the camera for a better view and the animation continues afterwards.

7.5.3 *Interface*

For the visual exploration of the pathlines, a graphical user interface must be provided. For this, buttons are employed in the style of standard media players. This includes a play, pause, forward, and backward button. Furthermore, a timeline is provided. The play button starts the animation, whereas the pause button stops the animation, but the current frame step is kept. The forward and backward buttons are useful to guide the animation step by step. Additionally, the possibility to jump to a certain keyframe is provided by clicking on the corresponding position on the timeline. This ensures an intuitive and fast exploration of the animation.

7.6 EVALUATION

This section is divided into two parts. In the first part, the novel surface shading was examined in comparison to other methods. An evaluation was conducted to assess the possibility to perceive surface information as well as flow data. In the second part, the pathline animation as well as the adapted surface highlights were evaluated. Here,



$\mathbf{ST}_1$ [66] $\mathbf{ST}_2$ $\mathbf{ST}_3$

Figure 66: The new shading approach ($\mathbf{ST}_3$) in comparison with semitransparent visualization ($\mathbf{ST}_1$) and Fresnel opacity ($\mathbf{ST}_2$) [66] for different vessels with internal flow depicted with illustrative streamlines. The semitransparent approach fails to give a spatial impression. The Fresnel opacity approach provides good visual results, but some streamlines are occluded by fully opaque vessel regions facing away from the viewer. The new approach ($\mathbf{ST}_3$) overcomes these limitations and the streamlines are depicted well.

the goal was to figure out the usefulness as well as the applicability to examine the flow in the medical and biomedical research stage.

7.6.1 *Evaluation of Surface Shading*

An informal evaluation was conducted. Here, three shading techniques were compared: semitransparency ($\mathbf{ST}_1$), fresnel opacity ($\mathbf{ST}_2$) according to Gasteiger et al. [66], and the novel approach ($\mathbf{ST}_3$), see Figure 66. The goal was to assess their capabilities for expressing rel-

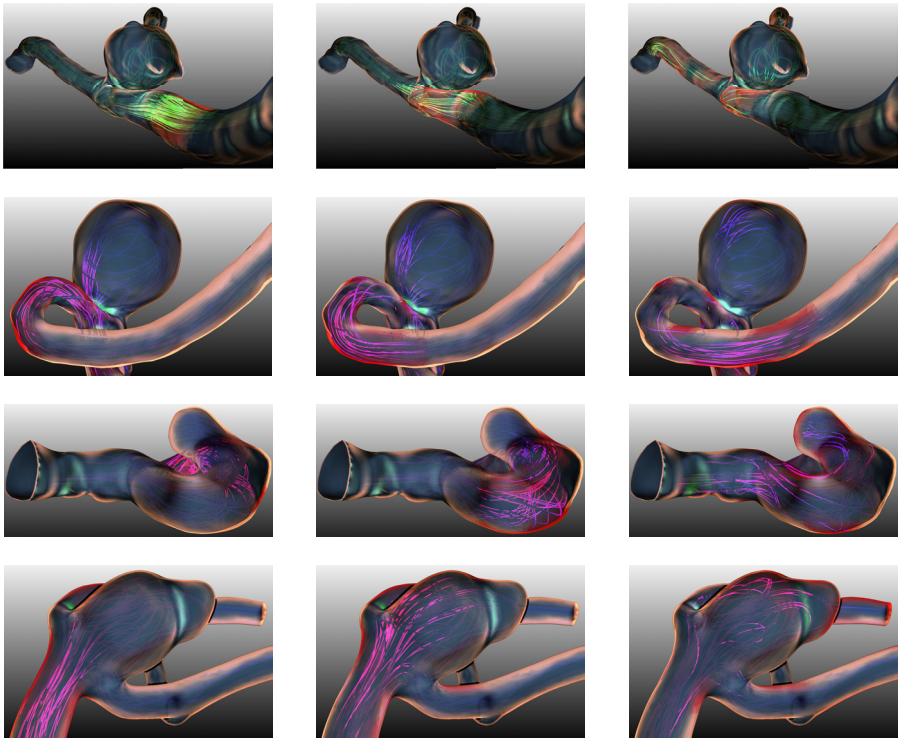


Figure 67: The novel adaptive surface approach ($\mathbf{ST}_3$) highlights pathline-nearby surface features during the animation of the pathlines. In the first row, a different color-coding for the pathlines than for the other datasets was used.

evant surface characteristics whilst simultaneously provide appropriate visibility of the embedded streamline visualization. Furthermore, the goal was to assess which of the proposed shading techniques yields the most expressive results. Therefore, an evaluation was conducted with one physician, two CFD engineers involved in hemodynamic analysis, and six researchers with background in medical visualization. Four representative vessel structures consisting of three cerebral aneurysms and one aorta dataset were chosen.

During the evaluation, the participants' spoken comments were noted and the participants were able to adjust the parameter settings for each technique, i.e., transparency value for $\mathbf{ST}_1$, edge fall-off parameter for $\mathbf{ST}_2$, and brightness value c for $\mathbf{ST}_3$ (recall Sec. 7.4.4). For each dataset and technique, the participants were asked to perform three tasks:

Task 1 Identification of salient surface features such as concave, convex regions and performance assessment of each shading technique to accomplish the task.

Task 2 Visibility assessment of the embedded streamlines.

Task 3 Assessment of spatial relationships and depth perception between vessel sections.

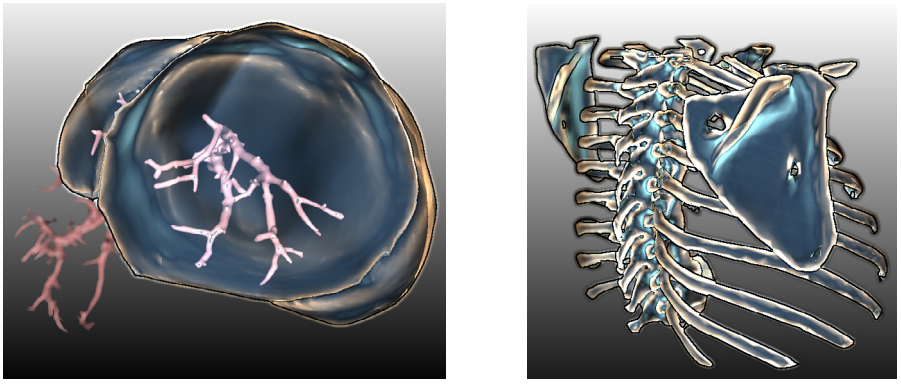
For *task 1*, technique $\mathbf{ST}_3$ was rated as most efficient and it revealed more surface features compared to $\mathbf{ST}_1$ and $\mathbf{ST}_2$. The participants stated that certain curvature features at branches and bulges on the aneurysm sac were more clearly depicted with $\mathbf{ST}_3$ than with $\mathbf{ST}_1$ and $\mathbf{ST}_2$ (see Fig. 66, third row). An increase of the opacity for $\mathbf{ST}_1$ improved the perception, but also increased the occlusion of the embedded streamlines. Therefore, technique $\mathbf{ST}_2$ was rated better than $\mathbf{ST}_1$ because an increased edge fall-off reveals more shape features but still ensures visibility of the flow facing towards the viewer. Most of the participants also appreciated the capability of $\mathbf{ST}_3$ to convey the salient surface features even in still images. For some vessel structures, the other two techniques required more camera interaction efforts to obtain an overview about the shape.

The assessment in *task 2* was rated most efficient for technique $\mathbf{ST}_3$ when using a brightness value around 1.5. Larger values would occlude more streamlines, which is similar to $\mathbf{ST}_2$ in terms of the edge fall-off value (in average 1.5). For this technique, some participants criticized the increased occlusion of surface parts facing away from the viewer. As expected for $\mathbf{ST}_1$, an increased transparency improves also the streamline visibility but decreases the shape depiction. Some participants stated that this assessment depends on the exploration task, i.e, focusing on embedded flow or on flow and enclosing vessel.

For *task 3*, $\mathbf{ST}_2$ and $\mathbf{ST}_3$ were rated as most efficient compared to $\mathbf{ST}_1$ because of the added shadow and depth cues. Thereby, blurring and depth attenuation were evaluated equally expressive with slightly more preference for blurring because of its more natural adaption to the human depth perception. Two participants also asked for a possibility to change the region of interest for the focus region by clicking on a specific vessel region.

7.6.2 *Informal Feedback on Nearby-Pathline Surface Highlighting*

An informal interview with a domain expert was performed to gain a qualitative user feedback. The expert is actively involved in the exploration of hemodynamics in cerebral aneurysms. The interview was designed to determine if the animation is useful and supports the expert in his diagnostic activities, such as the identification of slow and fast flow regions as well as near-wall flow. Therefore, four different datasets were shown, see Figure 67. The domain expert was asked to examine different aneurysms with pathline animation. It was first stated that it is useful to illustrate the pathlines over the whole time with desaturated color-coded information, e.g., flow speed and vorticity. The animations are visually pleasant and the exploration with step back and step forward is helpful. Furthermore, the expert ex-



(a) (b)

Figure 68: The novel adaptive surface approach can also be applied to non-vessel surfaces such as (a) liver surfaces with portal vein and (b) bone structures.

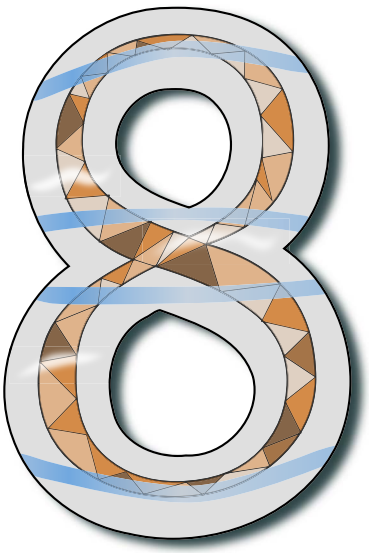
plained that the examination of near-wall hemodynamics is of high importance, since it can be an indicator for aneurysm rupture in the past and the increased risk for rupture. Thus, the highlighting is appropriate to assess the pathline distances to the surface. Here, less camera rotation is needed to estimate distances. The interruption of the animation during camera rotation was also stated to be very efficient to analyze specific regions. In summary, the domain expert was satisfied with the pathline animation and with the highlighting of nearby surface regions. Furthermore, the domain expert suggested to integrate different options for future work. For example, it would be useful to depict other information during the animation such as wall-shear stress. As the domain expert liked the surface shading, another suggestion is to decode the distances with another color bar. This would enhance the distance estimation.

7.7 CONCLUSION

In this chapter, a novel adaptive surface visualization technique for blood vessels with embedded flow information was proposed. The shading technique is based on the *suggestive contour* measure, which ensures both an appropriate depiction of relevant local surface features and the visibility of the embedded flow visualization. Therefore, it has the advantage that relevant local features on surfaces are depicted. Furthermore, depth blurring was incorporated to enhance the perception of depth and spatial relationships. The informal evaluation with domain experts demonstrated an improved shape perception compared to existing techniques whilst simultaneously ensuring appropriate visibility of the embedded flow visualization. Moreover, the novel approach is able to convey the salient surface features in still images, which also enables its utilization for documentation pur-

poses. Besides its application on vessels with embedded flow information, the approach is also applicable to other non-vessel surfaces such as liver surfaces with internal structures and bone structures, as shown in Figure 68.

Wall Thickness Visualization



This section is partly based on:

Sylvia Glasser, Kai Lawonn, Thomas Hoffmann,
Martin Skalej and Bernhard Preim
Combined Visualization of Wall Thickness and
Wall Shear Stress for the Evaluation of Aneurysms
IEEE Transactions on Visualization and Computer Graphics
2014 (to appear)

8

WALL THICKNESS VISUALIZATION

8.1 INTRODUCTION

The understanding of vascular diseases, such as coronary heart disease and aneurysms, benefits from an expressive simultaneous visualization of the vessel wall comprising the inner border (between lumen and vessel wall) and the outer border (between vessel wall and surrounding tissue) – which will be referred to as inner and outer wall. Vascular diseases are not only characterized by deposits that lead to vessel narrowing, but also by vascular wall remodeling. The remodeling is an early indicator of the coronary heart disease and important for aneurysm rupture risk analysis. While wide-spread imaging modalities only represent the inner vessel wall, recent developments in imaging technology enable the detection of the inner and the outer wall. An expressive visualization of the inner and outer vessel wall is a special instance of an embedded surface visualization, where the local distances between two flexible tube-like structures should be conveyed. In this chapter, a new technique for this problem is presented.

For an improved patient-individual risk rupture assessment and thus improved strategy (e.g., the decision if endovascular therapy, neurosurgical clipping or no therapy at all should be carried out), hemodynamic *and* wall morphology information are needed. Therefore, this chapter is focused on a visualization to convey wall thickness *WT* and the wall shear stress *WSS*. In particular, highlighting parts of the vessel wall where the *WT* and the simulated *WSS* values are in a certain range is of interest. Therefore, a comprehensive 3D visualization will be presented in this chapter. This approach comprises an experiment, where intravascular ultrasound (IVUS) is employed to probe a dissected saccular aneurysm phantom, which was modeled from a porcine kidney artery. Then, the wall thickness was extracted and the resulting 3D surface mesh was employed for CFD simulation to include hemodynamic information. This chapter can be summarized with the following contributions[1]:

- A simultaneous 3D visualization of the inner and outer vessel wall. The *WT* is conveyed via distance ribbons and adapted, view-dependent GPU shading techniques.

1 This section is based on the work 'Combined Visualization of Wall Thickness and Wall Shear Stress for the Evaluation of Cerebral Aneurysms', which arose in cooperation with Sylvia Glaßer. My major task was about the visualization of the underlying data.

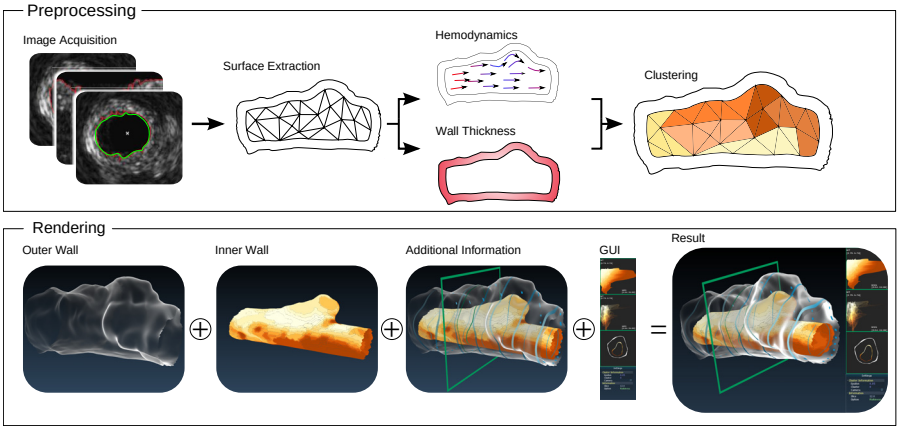


Figure 69: Overview of the approach that is divided into a preprocessing and a rendering step.

- Hemodynamic information (and a combination with *WT*) is provided via color-coding of the inner aneurysm wall surface.
- The visual exploration of the aneurysm model comprises brushing and linking (in the parameter space spanned by *WT* and *WSS*) as well as a surface clustering to provide surface clusters with similar parameter values on the inner vessel wall.
- Finally, a qualitative evaluation with domain experts indicates the value of the visualization and the strong need for wall morphology information.

8.2 RELATED WORK

In this section, the related work for the visual exploration of cardiovascular diseases and cerebral aneurysms with focus on the simultaneous depiction of outer and inner vessel wall is discussed.

The presented approach is also related to the visualization of embedded structures, which has been discussed in more detail in Chapter 7.

VISUAL EXPLORATION OF CARDIOVASCULAR DISEASES The simultaneous visualization of inner and outer vessel wall plays an important role for the evaluation of the coronary heart disease. Van Oijen et al. [200] reported a higher quality of direct volume rendering in comparison to surface rendering based on a contrast-enhanced computer tomography (CT) datasets. Hence, no wall thickness was extracted and their work aims at depicting the vessel lumen. Taking also the cardiovascular vessel wall into account, a direct volume rendering approach was presented in [73]. No inner and outer vessel wall was explicitly extracted, but the whole wall with pathologies like

artherosclerotic plaque was automatically highlighted with adapted transfer functions.

Balzani et al. [11] introduced a 3D reconstruction of geometrical models of atherosclerotic arteries (i.e., vessel walls with atherosclerotic plaque burden) based on multimodal image acquisition including IVUS, virtual histology data and angiographic X-ray images. The reconstructed 3D model comprises inner and outer wall. The outer wall is transparently rendered (without any additional information) and parameter values describing stress distributions are color-coded on the surface of the inner vessel wall. The visualization is combined with cross-sections that show the virtual histology data.

VISUAL EXPLORATION OF CEREBRAL ANEURYSMS The visualization of cerebral aneurysms generally focuses on the depiction of the lumen since no wall information is available. Higuera et al. [84] presented an automatic bidimensional transfer function approach for the direct volume rendering of aneurysms based on contrast-enhanced CT data. For the combined visualization with scalar hemodynamic information, cerebral aneurysms are usually displayed via color-coded surface views. In [33], the parameter *WSS* is mapped on the aneurysm surface. Neugebauer et al. [150] developed a 2D map display integrated into a 3D visualization of the relevant vascular structures for an interactive overview. Again, a scalar parameter is mapped on the surface. The visual exploration of blood flow gains importance due to its correlation with higher risk and more severe diseases. For the evaluation of the blood flow, color-coded streamlines, probe planes or glyphs are mostly employed within the application area of cerebral aneurysms [35].

BRUSHING AND LINKING This approach includes an interactive exploration concept comprising a scatterplot-based brushing and linking. Brushing and linking is well suited for the visual exploration and analysis of volume data [52]. A feature of the presented system that explicitly highlights the distance between outer and inner vessel wall was inspired by Dick et al. [47]. They presented two approaches for the interactive visualization of distances between two objects: cylindrical glyphs that smoothly adapt their shape and color-coding to varying distances during object movement and a set of slices that are color-coded.

SURFACE CLUSTERING Furthermore, the exploration of the data set is supported via surface clustering. The clustering on surface meshes most often aims at reducing the amount of triangles without a considerable loss of details. In contrast, the presented approach aims at the identification of connected points on the surface with similar parameter values. Therefore, a region merging method was employed.

Another technique is the region growing approach on surfaces, which is described in more general in [1]. The identification of certain clusters on mesh surfaces is also known as *mesh segmentation*. An application scenario is provided by the clustering and segmentation of protein surfaces [9]. A detailed overview about mesh segmentation approaches can be found in [4, 180].

8.3 IMAGE ACQUISITION AND PREPROCESSING

The approach can be divided into a preprocessing and a rendering step, see Figure 69. In this section, the preprocessing is explained, including the dissection and probing of the saccular artery aneurysm. Next, the extraction of the surface mesh, the *WT* and the hemodynamic information is explained. Finally, the surface clustering approach will be presented. Based on the extracted 3D model, a comprehensive 3D visualization was developed, which will be explained in more detail in Section 8.4. The visualization allows for assessment of the vessel's *WT* as well as the simultaneous evaluation of thickness and hemodynamic information like *WSS*.

8.3.1 *Dissection and Image Acquisition*

To overcome the missing in vivo imaging technique to depict the cerebral vessel wall, an artificial aneurysm from the artery of a porcine kidney of a dead pig was dissected, see Figure 70. The dissection was carried out within a clinical environment. An artery with a bifurcation was selected. Next, the smaller branch was shortened and closed. The preparation exhibits similar attributes like cerebral arteries, but can still be probed with IVUS.

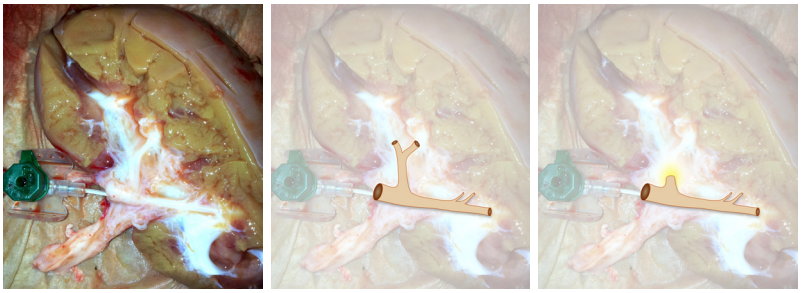


Figure 70: Preparation of the saccular aneurysm. Left, the porcine kidney is shown. In the middle, an artery with a bifurcation is dissected. Right, the artery branch was shortened of ca. 4 mm and closed yielding the aneurysm.

The dissected aneurysm was put on tubes and integrated into an artificial blood flow circle. Hence, no pumping was simulated. Next, IVUS was probed along the parent vessel with an IVUS system (Volcano Corp., San Diego, USA). During image acquisition, a saline so-

lution was injected, see Figure 71. A catheter was inserted into the vessel and pulled back along the parent vessel with constant velocity yielding a stack of 2D grayscale images that depict the vessel cross sections. The typical image parameters are: 512×512 pixels, IVUS diameter 20 mm and a pullback speed of 1 mm/s.

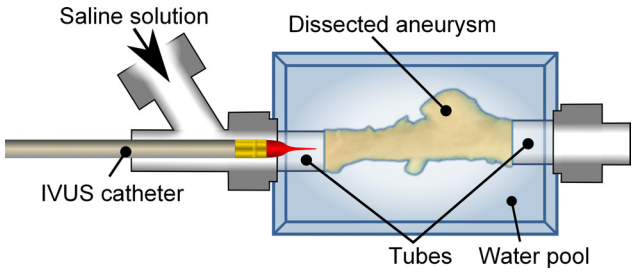


Figure 71: Probing of the aneurysm. During IVUS data acquisition, a saline solution was continuously injected (without pumping).

8.3.2 *Extraction of the Wall Thickness and the Hemodynamic Information*

Vessel wall detection algorithms as well as surface net generation approaches were adapted to the resulting IVUS dataset such that a 3D aneurysm model was created. For this purpose, a software prototype was developed. The postprocessing was inspired by [71] where the inner coronary artery wall was segmented based on the following steps:

1. Preprocessing of the IVUS image.
2. Transformation into polar coordinates.
3. A binary threshold segmentation to extract an initial line for the inner wall.
4. Iterative adaption of an active contour based on the initial line.
5. Extraction of the outer vessel wall by repeating steps 3 and 4.
6. *Optionally: One-click user interaction for additional parts.*
7. Combining inner and outer wall into a 3D surface mesh.

Preprocessing of the IVUS image. Due to the image acquisition, the catheter reflection is mapped in each vessel's cross section, see Figure 72(a). This reflection is masked out, and the boundary region around the circular mask is smoothed via Gaussian filtering to prevent sharp edges in the boundary region (see Fig. 72(b)). The distance marks are removed by replacing these predefined intensities by the mean values of their surrounding voxels. Due to the ultrasound inherent properties like low dynamic range, (blood) speckle and the low

signal-to-noise ratio [172], the whole image data is again smoothed with Gaussian filtering to reduce the high frequency noise. A 2D 3×3 kernel and a σ-value of 2.0 was employed, similar to the approach presented in [71]. In [172], more complex preprocessing methods and combinations for 3D ultrasound data are described, comprising speckle-removal methods for contour smoothing, median filters for gap closing in addition to the Gaussian filtering for noise reduction. In this case, the final segmentation result was sufficient based on the described filtering.

Polar coordinate transformation and extraction of the initial contour. A transformation into polar coordinates was carried out. Thus, the vessel walls can be detected as horizontal structures, see Figure 72(c)-(d). As proposed in [71], a binary threshold segmentation yields line segments of the initial contour. Hence, for each angle ϑ the voxel with smallest radius that exhibits a signal intensity larger than the threshold is selected.

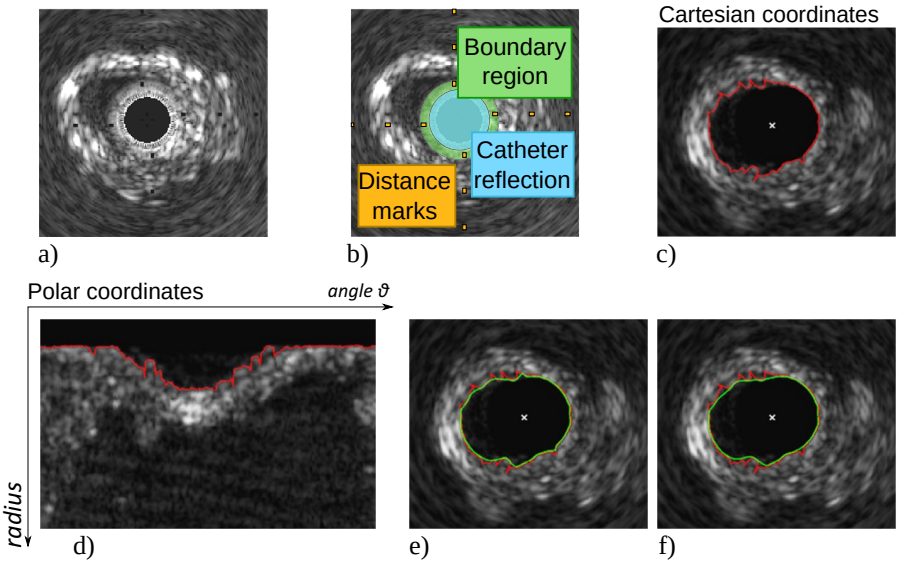


Figure 72: Preprocessing of the IVUS image data. In (a) and (b), the original dataset is depicted. The initial contour (red) extraction can be seen in (c) and (d). The iterative snake (green) adaption is illustrated in (e) and (f).Extraction of the inner vessel wall. First, the smoothed image is transformed into polar coordinates.

Iterative adaption of an active contour. In contrast to [71], where (based on the initial contour) ellipses were fitted to the IVUS images, cubic B-splines were employed to approximate the vessel wall. Hence, an aneurysm may exhibit an arbitrary morphology including varying cross sections. An active contour was iteratively adapted, also called *snake* [105]. The active contour segmentation is an established method in cardiology image analysis [198]. The snake is a parametrized curve $v(s)$, i.e., an energy-minimizing spline. It is influenced by image forces

that pull it towards features. The actual parametric position of a snake can be represented as:

$$v(s) = (x(s), y(s))^T \tag{37}$$

and its energy functional as:

$$E_{snake}(v(s)) = \int_0^1 E_{int}(v(s)) + E_{ext}(v(s))\, ds. \tag{38}$$

The term $E_{int}(v(s))$ refers to the internal spline energy, i.e., a first-order term and a second-order term, controlled by $\alpha(s)$ and $\beta(s)$:

$$E_{int}(v(s)) = \alpha(s)|\frac{dv}{ds}|^2 + \beta(s)|\frac{d^2v}{ds^2}|^2. \tag{39}$$

Hence, $\alpha(s)$ and $\beta(s)$ encode expectations concerning the smoothness and elasticity of the target structure's contour. Usually, they are constant. Hence, large values of $\alpha(s)$ increase the internal energy and allow the snake to stretch more and more. Also, large values of $\beta(s)$ allow the snake to develop more curves and will increase the internal energy.

The term $E_{ext}(v(s))$ refers to the external energy. It counteracts the inner energy and is derived by the grey values and the gradient of the image according to:

$$E_{ext}(v(s)) = w_1 f(x,y) - w_2|\nabla(G_\sigma(x,y) * f(x,y))|^2, \tag{40}$$

where w_1 and w_2 are weights, which represent the influence of the grey value $f(x,y)$ and the gradient $\nabla(G)$. The grey values are assumed to be normally distributed with the standard deviation σ. The initial snake was created as a parametrized spline based on the initial line (back-projected to Cartesian coordinates and connected to ensure a closed curve). The following weights were empirically determined: $\alpha = 0.3$, $\beta = 3.0$, $w_1 = 1$, $w_2 = 0.8$. Hence, small changes of these parameters did hardly influence the segmentation result. The iterative process terminates after no significant improvements were achieved. Since the extraction of inner and outer wall was carried out during preprocessing, the time consumption was not of interest. The resulting snake is depicted in Figure 72(f).

Extraction of the outer wall. The outer vessel wall was extracted in the same way. That means, the voxels with largest radius (see Fig. 72(d)) that exhibit a signal intensity larger than the threshold were selected and the process as described above is performed.

One-click user interaction for additional parts. With this step, a possible improvement of the segmentation result can be carried out. For the special case of overlapping structures, i.e., aneurysm parts that cannot be characterized by a single spline in an IVUS cross section view,

a one-click user interaction was included. Due to the small extent of the lumen in such areas, a primitive region growing algorithm was employed first and the border of the extracted region was used to adapt a second snake, see Figure 73(b).

Combining inner and outer wall into 3D surface mesh. The par-

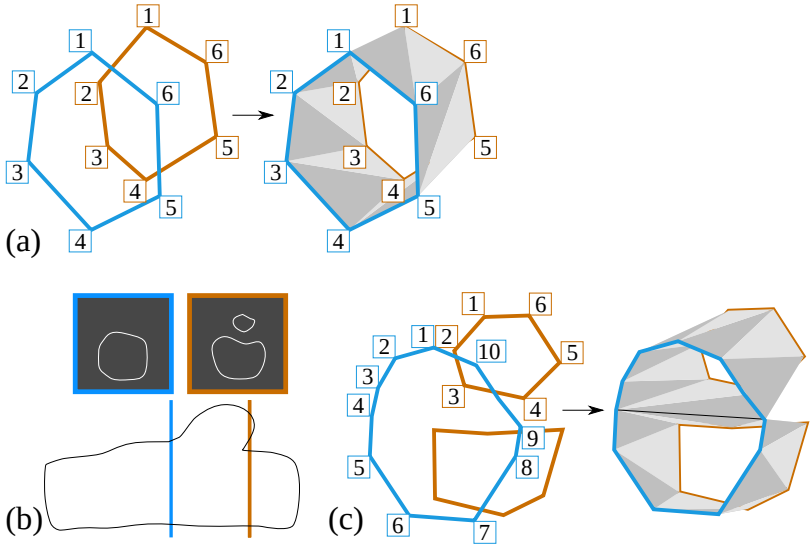


Figure 73: Triangulation of segmented contours. In (a), the triangulation based on two snakes from two subsequent image slices is illustrated. In case of additional contours due to overlapping structures (b), the first spline is sampled with $2n-2$ points instead of n points. Then, triangulation is carried out as illustrated in (c).

ameterized splines were employed to create two 3D surface meshes for the inner and outer wall. That means, for two subsequent contours, the discrete positions ($n = 128$) extracted from the splines are joined with triangles, see Figure 73(a). If a slice of the IVUS image holds two contours (due to the optional manual addition of overlapping aneurysm parts), ca. twice (2n-2) as much sample points are extracted from the previous slice and are employed for triangulation (see Fig. 73(c)). The triangulation from n to 2n-2 sample points is carried out by adding additional triangles. If the current slice holds one contour and the previous slice holds two contours, i.e., the overlapping part stopped, the contour with larger distance is triangulated, i.e., filled with triangles itself. The described problem is a special case of the branching problem, see [141] for more information. Due to the variable spline parametrization, the triangulation could be easily adapted.

Extraction of the wall thickness. Based on the initial 3D surface meshes comprising inner and outer wall, an advancing front approach is applied to improve the mesh quality via edge collapses and edge flips [178]. Building on this, the parameter wall thickness WT for each

point on the inner surface mesh as minimum possible distance to the points of the outer surface mesh and vice versa was extracted.

Extraction of hemodynamic information. The hemodynamic information was gained via CFD simulation based on the optimized inner wall surface mesh. Hence, an unstructured volume mesh was extracted and the blood flow was modeled as incompressible Newtonian fluid with rigid walls [34]. The presented framework focused on the *WSS*.

8.3.3 *Surface Clustering*

To support the subsequent visual exploration and analysis of the extracted aneurysm, all points of the inner surface mesh were clustered. A bottom-up hierarchical clustering approach was employed where only neighbored clusters could be merged into a new cluster (a process also known as *region merging*). Cluster c_i and cluster c_j are neighbored, if c_i contains a point that is connected to a point of cluster c_j via an edge on the 3D mesh surface. Initially, each point forms a single cluster. Then, the two neighbored clusters with the overall best similarity are iteratively merged into a new cluster. If the best similarity exceeds a certain threshold ϵ, the region merging terminates. The *risk* vector $\overrightarrow{r_i}$ for a vertex i is defined as:

$$\overrightarrow{r_i} = \begin{pmatrix} WSS_i \\ 1 - WT_i \end{pmatrix} \tag{41}$$

For the extraction of $\overrightarrow{r_i}$, both parameter spaces are globally normalized first such that their values are in the interval [0,1]. Thus, points on the surface with low *WT* and increased *WSS* yield larger values for $\|\overrightarrow{r}\|$ than points with a local thick wall and low *WSS* values. The definition of the used risk attribute was approved by the medical experts who evaluated the presented framework.

The similarity sim_{ij} between clusters c_i and c_j is extracted as

$$sim_{ij} = \|\overrightarrow{R_i} - \overrightarrow{R_j}\|, \tag{42}$$

where $\overrightarrow{R_i}$ is defined as the average values of *WSS* and $(1 - WT)$ for clusters c_i, precisely:

$$\overrightarrow{R_i} = \frac{\int_{c_i} \overrightarrow{r}\, d\mathbf{x}}{\int_{c_i} d\mathbf{x}}.$$

To support various divisions of the surface into clusters, the clustering result were precomputed for $\epsilon \in \{0.05, 0.1, 0.15, 0.2, 0.25\}$. During clustering, all regions are stored in a region adjacency graph.

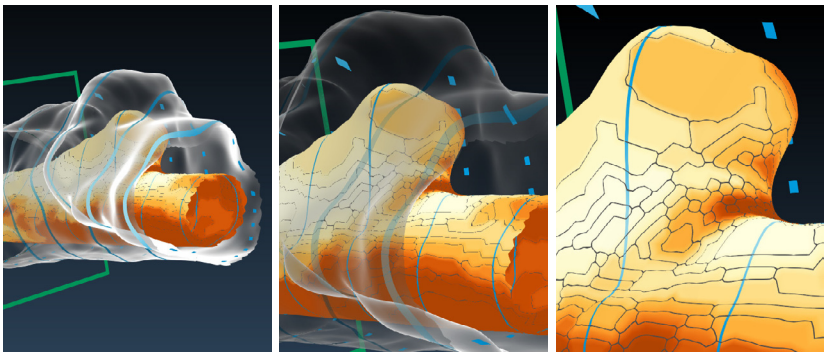


Figure 74: Illustration of the fading effect of the outer vessel wall visualization. From left to right, the distance to the view point is decreased yielding a fully transparently rendered outer wall.

8.4 VISUALIZATION FRAMEWORK

In this section, the different visualization and exploration techniques are described that are included in the framework. As depicted in Figure 69, the rendering step comprises the shading of the outer wall (explained in Sect. 8.4.1) and the color-coding of the inner wall's surface mesh (explained in Sect. 8.4.2). Hence, the visualization is adapted such that occlusions are avoided. The third part describes the exploration of the 3D scene in more detail (Sect. 8.4.3).

8.4.1 *Visualization of the Outer Wall*

The outer wall is displayed with a ghosted-view approach based on the Fresnel shading according to [66]. They employed a Fresnel-reflection model [177] and mapped this to the opacity. The opacity o is determined by $o = 1 - |\langle \mathbf{v}, \mathbf{n} \rangle|^r$, where $r \geqslant 0$ is the edge fall-off parameter. As a standard setting, $r = 1.5$ is used. With significantly less values, the outer wall would disappear. Higher values would interfere with the visibility of the inner wall. As this information is only important for gaining an impression on the thickness, a white color is used. This avoids a distraction of mixed colors when focusing on the inner wall. Hence, the color of the outer wall is determined by $\text{color}_{\text{Outer}} = \mathbf{1} \cdot o$, where $\mathbf{1} = \begin{pmatrix} 1 & 1 & 1 & 1 \end{pmatrix}$ and represents the RGBA values. If the user is interested in certain areas of the inner wall and zooms to this area, the outer wall may disturb the systematic investigation. To prevent this effect, a fade-out is implemented. The opacity of the outer wall fragments depends on the distance to the view point. Thus, the final color is determined by:

$$\text{color}_{\text{Outer}} = \mathbf{1} \cdot o \cdot \text{dist}\,.$$

The distance dist is measured in camera space. Therefore, a slight fading effect guarantees that the outer wall disappears when the view point is close, see Figure 74.

8.4.2 *Visualization of the Inner Wall*

In contrast, for the inner wall visualization, a color-coding with respect to a selected parameter (*WT*, *WSS* or $||\overrightarrow{\tau}||$) is used. A white to brown colormap is used and the parameter space is scaled into the interval $[0, 1]$. Then, a color map in $[0, 1]$ is employed for the inner surface mesh's points. Here, high values are represented by a brownish color, whereas low values correspond to white, see Figure 75.

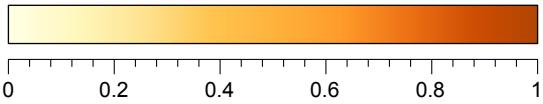


Figure 75: The employed colormap for color-coding.

Beyond the direct representation of each point's parameter values, a surface clustering was employed to provide surface clusters (recall Sect. 8.3.3). For the color-coding of the clusters, the surface integral S for each cluster c_i of the riskiness $||\overrightarrow{\tau}||$ is determined:

$$S_i = \frac{\int_{c_i} ||\overrightarrow{\tau}|| \, d\mathbf{x}}{\int_{c_i} d\mathbf{x}}.$$

As the value of S_i lies in the interval $[0, 1]$ for each cluster c_i, the colormap from $[0, 1]$ is assigned to every cluster (Fig. 75). Thus, every fragment of the inner wall is colored according to its associated cluster region. For a better differentiation of adjoined cluster regions, a border is visualized additionally. As the underlying vessel structure is a triangulated surface mesh, three cases may occur (see Fig. 76). First, all triangle points belong to the same cluster and the triangle is rendered with the assigned cluster's color. In the second case, two of three points in a triangle share the same surface cluster and in the third case all points are associated to different clusters. For the second case, the border is created by connecting the midpoints of the lines that are incident to the point of the divergent cluster. For the last case, the centroid is connected to the midpoint of the edges. As a result, the presented representation of the outer wall serves as context object and the inner wall with its surface clusters is the focus object.

8.4.3 *Data Analysis*

The presented framework is depicted in Figure 77. In the center, the 3D view of the outer and the inner wall is presented. Based on in-depth discussions with the clinical experts, an additional techniques

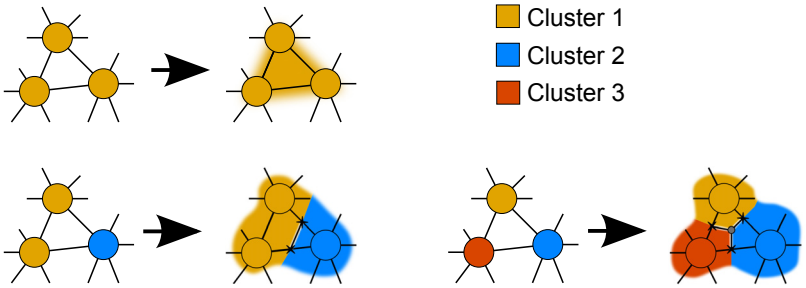


Figure 76: Adaption of the rendering to depict the clustering result.

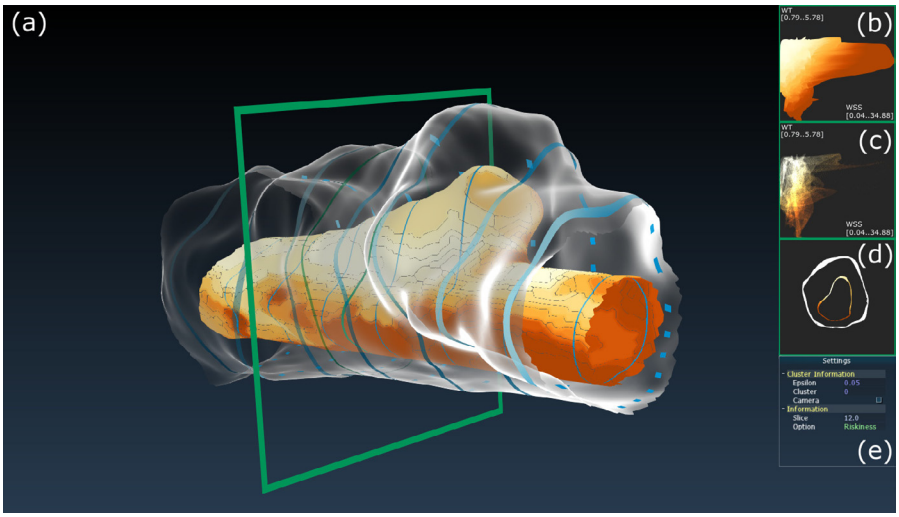


Figure 77: Presentation of the presented framework for the exploration of aneurysm wall thickness and wall shear stress. The main area (a), holds the visualization of the inner and the outer wall as well as distance ribbons. On the top right, the global scatterplot (b) and the local scatterplot (c) are depicted. The slice view (d) depicts the vessel wall's cross section. Its position is marked with the green rectangle in (a). Interactive brushing and linking is carried out in the global scatterplot and parameter choices can be set in the user panel (e).

was provided to explore the 3D visualization of the vascular surfaces. First, the estimation of *WT* is supported by adaption of distance ribbons, which will be explained in more detail. Second, the global and local scatterplot was developed (recall Fig. 77), including a brushing and linking facility for the interactive exploration of the data. Furthermore, the surface clustering can be employed for a cluster-based exploration. Finally, the slice view conveys cross-sectional views of the inner and outer wall.

Visualization of the parameter *WT* with distance ribbons. In general, the *WT* could be visualized via color-coding. However, in the presented framework color-coding is already employed to depict the surface clusters. Dick et al. [47] presented different ways to illustrate

distances of inner and outer structures – in their case a bone and an implant. To convey the thickness, *distance ribbons* are used. Similar to slices, which were used by Dick et al., the distance ribbons wrap the inner and outer structure. They are color-coded from dark blue to bright blue depending on whether the distances are small or high. Additionally, the distance ribbons on the back side of the outer wall are dashed to impart a spatial impression of the surface, recall Figure 77. The distance ribbons are well suited for conveying distances in a static image.

Global scatterplot. The *global scatterplot* is used to display the *WT* and the *WSS* of the inner wall, which is divided into different surface clusters visually represented by different colors (recall Sect. 8.3.3). According to their association in the surface cluster, the corresponding point in the global scatterplot is identically colored. The global scatterplot provides an overview of the underlying data of the vessel. For the specific examinations of data in the global scatterplot, a brushing and linking approach is integrated. The user can brush peculiarities in the global scatterplot and this accentuates the region in the global scatterplot as well as the corresponding parts of the inner wall. If he is interested in, e.g., regions with a high *WSS* and medium *WT*, he can brush this specific polygonal region in the global scatterplot and the corresponding region parts are emphasized with a different colormap, see Figure 78. This greatly supports the visual exploration as well as the communication and the sharing of knowledge if two medical experts discuss about the data. An eraser tool is also provided to deselect parts of the brushed region.

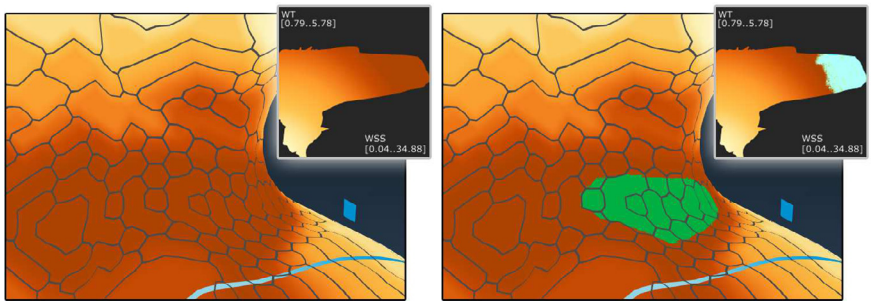


Figure 78: If the user is interested in regions of high *WSS* and medium *WT*, he can brush a polygonal region in the global scatterplot (inlet) and the region will be emphasized with a different colormap from green to white.

Local scatterplot. Furthermore, a *local scatterplot* is employed as well. Here, only the data that can be seen in the current 3D scene are plotted. This is an advantageous technique to support local examinations of the vessel. If the medical expert is interested in analyzing specific regions, the local scatterplot allows representing the data of the local observation, see Figure 79. Each fragment in the 3D scene plots its specific data values to the local scatterplot.

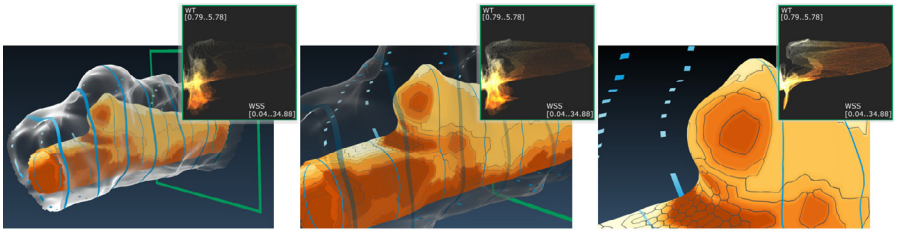


Figure 79: The local scatterplot (inlet) adapts its appearance according to the specific 3D scene. Therefore, the expert gains an insight about the data distribution in the current scene.

The slice view. For assessing the thickness, a slice view is provided. A green frame is used in the 3D scene (see Fig. 77(a)), which can be translated along the vessel. Simultaneously, the silhouette of the inner and outer wall is illustrated in the slab view, see Figure 77(d). The parts of the inner wall are at this juncture color-coded according to their cluster representation. Furthermore, a slice is provided in the 3D scene for the orientation. The slice is also depicted on the inner and outer wall for a better illustration of the current position.

Visual exploration of the surface clustering. As described in Section 8.3.3, different ϵ values are used to obtain different cluster results. Therefore, the user can select the ϵ values to investigate the results. Furthermore, the user can also choose specific clusters. To support medical experts, the clusters are ranked according to their average riskiness value. If the expert selects one cluster, only this cluster is color-coded on the surface whereas all other surface parts are shaded in grey. If the cluster is currently hidden, it is gradually made visible. For this purpose, an automatic camera path from the current view point to a destination where the camera points to the selected cluster's center is provided. This concept was inspired by [229], where similar camera paths were successfully employed and appreciated by the medial experts.

8.5 GPU-IMPLEMENTATION

In this section, the rendering part of the presented framework is described in more detail regarding the GPU-implementation.

The surface visualization is divided in a part for the inner wall and a part for the outer wall. The outer wall is conveyed using a Fresnel shading term. The inner wall is color-coded according to the selected parameter space or to the cluster's average parameter. The colorings as well as the corresponding borders are generated in the fragment shader. Therefore, the different cases which may occur will be tested in the geometry shader. If two vertices have the same cluster ID but not the third vertex, the two vertices are assigned to -1 and the third vertex is assigned to 1. These values are interpolated on the trian-

gle in the fragment shader and therefore the fragment is assigned to a grey value if the absolute value does not exceed a specific border length. This ensures a border in a triangle separating the two clusters. If all vertices correspond to different surface clusters, every vertex is assigned to the unit vector $(1,0,0),(0,1,0),(0,0,1)$. The fragment shader interpolates these vectors such that the resulting vector corresponds to the barycentric coordinates of the original triangle. Afterwards, the three lines are determined that connect the barycenter of the triangle with the midpoints of the line segments. Finally, the distance of barycentric coordinates to the lines are tested. If the distance deviates less than the border length and the orthogonal projection is between the barycenter and the midpoint of the line, the fragment will be colored grey.

For the illustration of the scatterplots and the slice view, the extension EXT_shader_image_load_store is used. This extension allows drawing on specific coordinates on an image. Both scatterplots are initialized as images. The global scatterplot is drawn in an extra shader. Here, all *WSS* and *WT* data are available. These values are scaled according to the image size and plotted on the global scatterplot. The local scatterplot is generated in the fragment shader of the surface visualization. Hence, only fragments are drawn, which can be seen in the 3D scene. Thus, the local scatterplot only depends on the fragments of the current 3D scene. The corresponding fragments inherit *WSS* and *WT* information. Thus, every fragment stores its *WSS* and *WT* in the image of the local scatterplot.

The slice view is also generated in an additional shader as slab rendering. A global variable is used to specify the position of the slice view. The shader has also vertex position information. The vertex position will be send to the fragment shader independently whether this can been seen in the current view. The fragment shader tests if the 3D coordinates are in the range of the position of the slice view. In this case, the position is drawn on the image of the slice slab.

The camera path is determined by using the center of each cluster. With the center's normal, the camera's destination position is determined. Then, a path is extracted from the current camera position to the end point. Furthermore, the center's normal is employed for slightly changing the camera's view direction.

8.6 EVALUATION

For an evaluation of the combined visualization of *WT* and *WSS*, a questionnaire was prepared. Eleven subjects (one female, ten male, aged 26-41) participated in the user study, comprising two experienced neuroradiologists and nine biomedical engineers familiar with vascular diagnosis. The user study began with a short demonstration of the framework and a description of different viewing techniques.

Afterwards, each facility was explained in more detail and the users were encouraged to explore the scene on their own in order to answer the questions. The spoken comments of the participants were noted.

8.6.1 *Questionnaire*

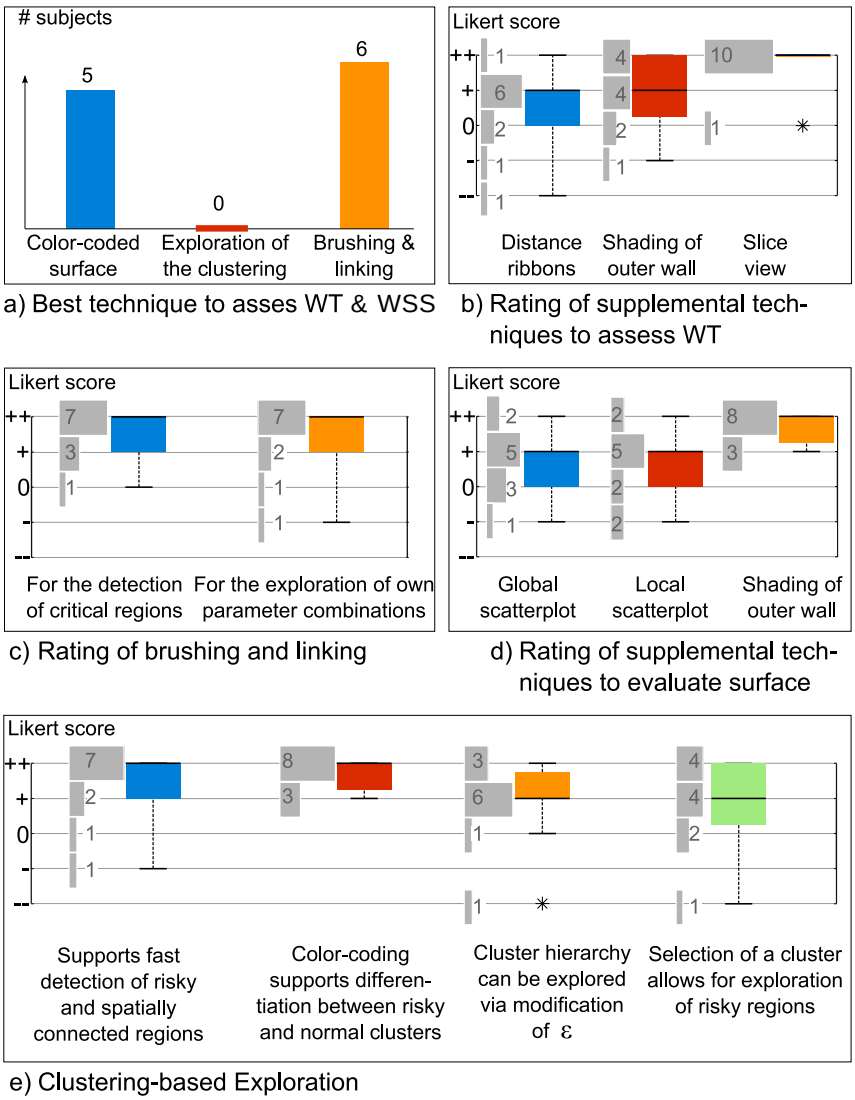


Figure 80: Selected results of the user study based on the questionnaire. The box-whisker-plots depict the lower q1 (25%) and upper q3 (75%) quartile as well as the median, min and max value. Histograms depict the complete ranking. The extreme values are considered to be outliers (*) if they are at least two interquartile ranges below q1, or at least two interquartile ranges above q3.

For all questions, pre-defined choices were provided to make the results comparable. Besides simple "yes" or "no" and multiple choice questions, a Likert-type scale [127] was employed to rate the suitability of selected techniques. That means, the user can select between

the choices - -, -, o, +, ++. Then a score $\in$ [-2,2] was extracted. The questionnaire comprises the following aspects.

General aspects. First, the subjects were asked about their opinion about the importance of the wall thickness for the evaluation of cerebral aneurysms in general. They should also state, if they would use the framework. Another question was about which of the techniques (general exploration of color-coded surface view of inner wall, exploration via clustering, exploration via brushing and linking) is best suited for assessing the wall thickness. Hence, the intent was to assess which is their favorite exploration technique. Finally, the subject should rate the provided distance ribbons, the shaded outer wall, and the slice view regarding their ability to depict the *WT*. The last techniques were declared as supplemental techniques.

Brushing and linking. The second aspect aims at the suitability of the scatterplot-based exploration and the brushing and linking facilities. The users rated the ability to detect critical regions via brushing and linking. Second, they rated the possibility to brush their own parameter combinations, e.g., regions with low *WSS* and low *WT*. Furthermore, they chose how well the distribution of *WSS* and *WT* can be extracted from the global and the local scatterplot.

Clustering. For the presented clustering view, the users should rate if the color-coded clusters are appropriate to identify spatially connected regions with increased $\|\overrightarrow{\tau}\|$. They could choose the Likert score "- -" if they do not like the clusters at all but prefer a direct color-coding of parameters on the inner wall surface. Also, they should rate if the color-coding is well suited to separate regions with high risk from regions with low risk. Next, the subjects rated the ϵ-based exploration of the clustering results. Finally, they evaluated the selection of single clusters to detect dangerous regions. They should also assess, if various datasets could be compared based on the clustering, i.e., small and many regions indicates a more heterogeneous distribution of *WT* and *WSS*.

Navigation and general remarks. Two questions are related to the navigation. First, the complexity of the necessary navigation to inspect and explore the whole scene was requested. Second, the suitability of the automatic camera path animation had to be rated. Finally, the participants were asked for additional remarks or possible extensions of the presented framework.

8.6.2 *Results*

For interpretation of the pre-defined Likert score answer categories, the mode value m, i.e., the most frequent answer, and the percent aged amount P of participants who chose "+" or "++" was provided. All users assigned a high importance of wall thickness evaluation for treatment planning of cerebral aneurysms (m = "++"; P = 100%) and would employ the presented framework. When choosing their favorite facility to explore the parameters *WT* and *WSS* between color-coded surface visualization, cluster exploration and brushing and linking, the participants chose the brushing and linking (6 of 11) and the color-coded surface visualization (5 of 11), see also Figure 80(a). Brushing and linking was chosen due to the direct visual feedback, i.e., the interactive highlighting of surface parts corresponding to the brushed region in the scatter plot. When rating the supporting visualization techniques, the slice view was ranked best with m = "+"/ "++" and P = 91% (see Fig. 80(b)). All users attested a very easy navigation (m = "++"; P = 100% for the usability of navigation and the automatic camera paths) since all of them could easily explore the scene in an intuitive way.

The exploration via the brushing and linking methods was substantial for evaluating the 3D aneurysm model and the users most often rated it with m = "++"; P = 91%. They also rated the possibility to detect own parameter combinations via brushing and linking with m = "++"(P = 82%), see Figure 80(c). Especially the medical experts pointed out that a ground truth for the most dangerous parameter combination is still missing. Although they agreed with the pre-defined combination of *WT* and *WSS* as riskiness $\overrightarrow{r}$, they stated that for some aneurysm characteristics (e.g., monitoring during longitudinal studies) a modified parameter combination is needed for exploration. Compared to the brushing and linking facility, not all users appreciated the exploration based on the global and local scatterplot (m = "+"; P = 64% for both). They named the missing spatial information as shortcoming of the scatterplot-based exploration. The users liked the adaptive fading of the outer wall visualization very much (m = "++"; P = 100%), see Figure 80(d). Hence, the users were not asked to compare the techniques, but rather rate each of them individually.

The clustering-based exploration enabled the fast detection of dangerous, spatially connected regions (m = "++"; P = 82%) and the color-coding favored a fast separation into normal regions of the inner wall and regions with high riskiness (m = "++"; P = 100%). Hence, a variation between usefulness of the selection of individual clusters (m = "+"/ "++" and P = 73%) and the exploration of the clustering's hierarchy by varying ϵ (m = "+" and P = 82%) could be seen, since some participants did not want to use the automatic division into

clusters (see also Fig. 80(e)). Interestingly, none of these participants were clinical experts. The clinical experts approved the clustering-based exploration (selective m = "+"). When asked for a Likert score regarding the possibility of comparison different datasets based on the clustering results, the users answered with m = "+"/ "++" and P = 100%.

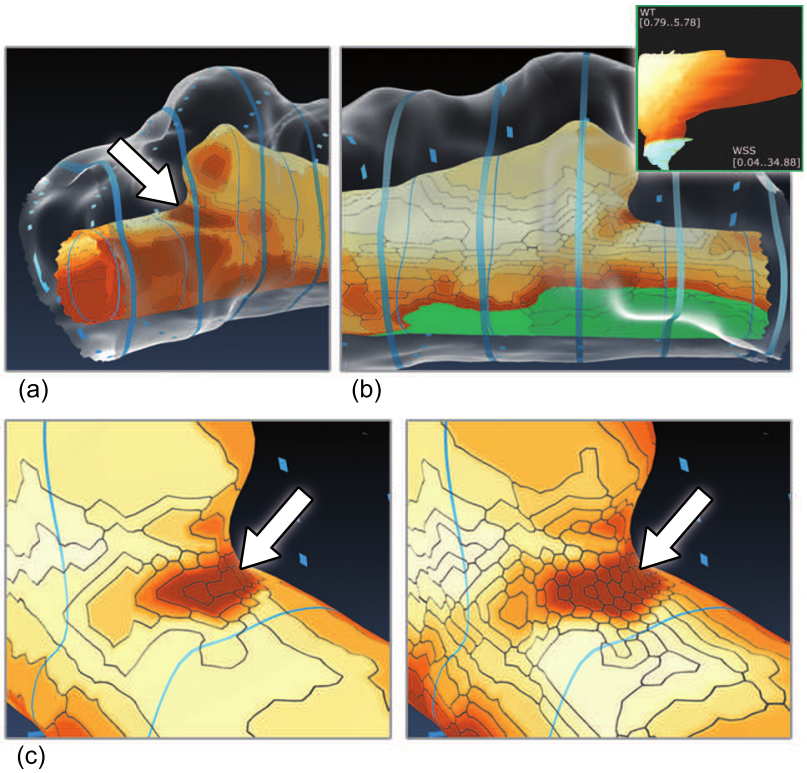


Figure 81: Selected aspects during the user exploration. In (a), the salient region is shown. In (b), an important finding is visualized. During the brushing of a user-defined parameter combination, the highlighted regions at the bottom of the parent vessel were identified. In (c), different cluster results (based on varying ϵ values) still reveal the salient region.

8.7 DISCUSSION

The informal evaluation and especially the discussion with the medical experts indicated the importance of *WT* for aneurysm evaluation. The participants also stated that wall morphology and thickness is one of the most important missing information for rupture risk (due to the missing in vivo imaging technique) based on their clinical experience. In general, the users would definitely employ the presented framework for simultaneous evaluation of *WSS* and *WT*. During evaluation, two main techniques were discussed in depth: the brushing and linking and the clustering. These aspects will be described in the following. Next, minor improvements and suggestions are listed. Fi-

nally, general aspects regarding the image acquisition are discussed and a short discussion for another application area is provided to demonstrate the practicalness of the presented method.

Brushing and linking. Most users chose the brushing and linking-based exploration since they can also highlight own parameter combinations (see Fig. 80(a)). Especially the facility to explore own parameter combinations was rated as useful. Hence, the clinical experts referenced the missing gold standard of rupture risk parameters, recall Section 7.2. During exploration, all participants explored the regions with high *WSS* and low *WT* at the most salient surface part, see Figure 81(a). As pointed out by one user, an interesting feature was detected at the bottom of the aneurysm parent vessel, see Figure 81(b). Hence, a thin wall and low *WSS* values were brushed yielding the highlighted surface parts. In clinical practice, these regions are important to completely characterize the relationship between vessel topology and hemodynamic behavior. The clinicians stated that also areas with wall thinning (and thus increased rupture risk) can be a consequence of reduced blood flow and lower but oscillating *WSS* values (recall Section 7.2). A major feature was requested by one of the medical experts. He asked for a reversed brushing and linking concept. That means, a brushing in the dataspace (on the inner vessel wall surface) causes a linked highlighting in the attribute space (the local and global scatterplots). Although this relationship can be explored via cluster selection (selected cluster is highlighted on the surface, as well as its parameter values in the scatterplots) a brushing and linking concept in this domain seems to be interesting. In particular, the intention was to adapt the concept presented in [115]. They employ a brushing facility on a surface for annotation purposes in a GPU-based virtual endoscopy environment.

Clustering-based exploration. The pre-extracted clustering results were discussed controversially. On the one hand, many users (i.e., 8 of 11) liked the clustering as well as the ϵ-based exploration of the hierarchical clustering structure (recall Sect.8.4.3). That means, with smaller ϵ values, the region merging process is stopped earlier. Therefore, more clusters at heterogeneous regions, i.e., regions with strong variations for the parameter values, will remain and not be merged, see Figure 81(c). The users did not request a clustering based on different parameter combinations, but instead combined it with the brushing and linking technique for this purpose. The users that did not like the clustering, explained that they just do not get any benefit from the clustering. Hence, it is suggested to use the direct color-coding of one of the parameters *WT*, *WSS* or $\|\overrightarrow{\tau}\|$. Here, the clustering visualization is purely supplemental. At the moment, it is not aiming at any automatic parameter extraction or classification methods.

Minor aspects. After the evaluation, minor improvements were requested by the users. For example, the inclusion of measurement tools to approximate the *WT* was suggested. Also, they asked for additional quantitative information about the cluster's average parameter values. Furthermore, a user suggested distance ribbons with varying width adapted to the values of *WT*. This would lead to very broad ribbons in areas of thicker walls and induce occlusion problems.

General aspects regarding image acquisition and segmentation. General aspects that can be improved are related to the (preprocessing) pipeline. First, the dissected aneurysm was probed with IVUS. Recently, optical coherence tomography yields promising results in intravascular coronary imaging and a better image resolution than IVUS, see [97]. However, similar to IVUS, no admission for the in vivo imaging of cerebral vascular system is available yet. Second, the presented snake-based segmentation works fine for the prepared aneurysm, but more approaches, e.g., level sets, exist. The level set method was adapted as well, but empirically it achieved worse results compared to the snake-based segmentation. In [10], a comparison of different IVUS 2D and 3D segmentation techniques is provided. Also, a co-registration with Bi-plane angiography image data is possible. Due to the presented setup, i.e., the parent vessel was fixated along the z-axis, this step was not necessary.

Outlook. Naturally, the presented framework can be adapted to vari-

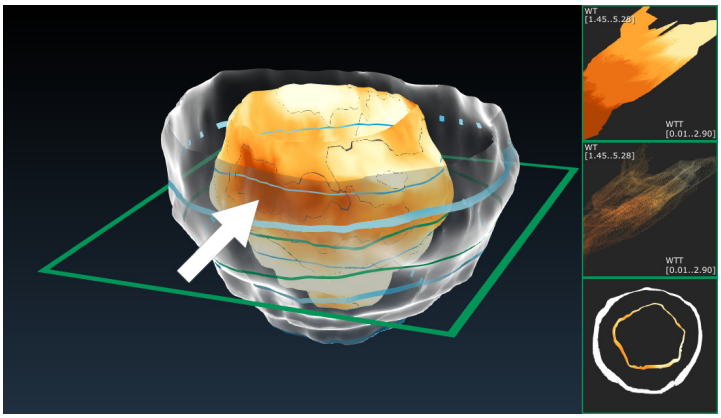


Figure 82: Adaption of the presented framework to assess myocardial infarctions. Hence, the myocardial contractility is analysed to detect regions on the myocardial surface with reduced wall thickening from end diastole to end systole. Parts of the myocardium is characterized by reduced wall thickening due to a heart attack (see arrow).

ous application areas comprising cerebral aneurysms as well as inner and outer vessel walls. Beyond these vessel wall concepts, scenarios

comprising a surface enclosing a focus structure yield an important application area. Therefore, a study of functional MRI datasets of the left ventricle, i.e., the heart muscle also called myocardium, from patients that suffered from a heart attack was examined. Although contrast-enhanced datasets are acquired to locate the infarction scar and the myocardial viability, the functional MRI data holds valuable information about the heart muscles contractility. Typically, the infarction causes a reduced ability of muscle tension. Hence, the clinical expert does not only want to locate the infarction scar and necrotic tissue, but he also wants to assess the influence of the infarction for the myocardial function. For clinical evaluation, the heart cycle is analyzed from the time of end diastole (the myocardium is fully relaxed) until end systole (maximum contraction is achieved). In Figure 82, the adaption of the framework to this application scenario is demonstrated. The left ventricle consisting of epicardium and endocardium, i.e., the myocardium, is visualized with the outer wall shading and the color-coded surface view. The myocardial contractility is extracted as wall thickening. Therefore, the wall thickness at end diastole and end systole is approximated. The wall thickening is then calculated as increased wall thickness at end systole and color-coded with the presented colormap.

8.8 CONCLUSION AND FUTURE WORK

In this chapter, a framework for the simultaneous exploration of hemodynamic information, i.e., the *WSS*, and the wall thickness parameter *WT* was presented. This work is a first step towards the integration of the vessel wall morphology in the complex image-based evaluation of cerebral aneurysms and similar vascular diseases. More generally, the presented visualization and exploration concepts can be applied for the inner and out vessel wall and additional information, like the extracted *WSS*. The presented framework depicts the outer and inner wall and avoids occlusions. Color-coding is employed on the inner vessel wall surface to depict (cluster-based) parameter values. Furthermore, the exploration is improved with a global and a local scatterplot, (including brushing and linking facilities) as well as a slice view and the clustering-based exploration. The presented framework is the first framework that provides a combined visualization of wall thickness and hemodynamic information for a dissected cerebral aneurysm.

Due to the novelty of this research area, many extensions are possible. The most interesting one (from the clinical point of view) would be an improved CFD simulation that accounts for the vessel wall thickness and its elastographic properties. Also, the influence of different image acquisition techniques, like optical coherence tomography [95], will be examined in the near future. For instance, improved

image modalities, i.e., higher resolution image datasets, will not only provide the vessel wall thickness but also the vessel walls pathologic parts, e.g., plaque burden, calcified parts of the vessel wall as well as its elastographic properties.

The next step is a carefully adapted depiction of the presented visualization and exploration techniques of streamlines to reveal additional information like hemodynamic vortices or increased inflow jets.

Part IV

CONCLUSION

Conclusion 9

9 CONCLUSION

9.1 SUMMARY

Illustrative visualization has many application areas and is useful for various kinds of problems. This thesis showed different fields of illustrative visualization concepts and provided several improvements. Therefore, several state-of-the-art methods were discussed and different disadvantages were identified. Motivated by these drawbacks, a few illustrative visualization techniques for selected visualization problems were developed.

This thesis was roughly divided into three parts. Part i presented the differential geometry, the discrete differential geometry, and an application of curve smoothing on triangulated surfaces. The necessary tools for developing and understanding the main concepts were presented. An insight into the differential geometry with all important requirements for understanding the further line drawing techniques was given. Afterwards, the differential geometry was extended to the field of discrete differential geometry. Thus, terms like gradient or curvature can be approximated on triangulate surfaces, which is crucial for the development of algorithms that generate visually pleasant illustrations. Finally, this part closed with a first application of curve smoothing on triangulated surfaces. Here, an acquisition pipeline was provided where medical image data, in this case vessels with interior blood flow, was assessed for further tasks. Afterwards, a method was introduced which smoothes initially drawn surfaces curves. Finally, the user may cut the surface along this curve to extract regions of interest or to delete unessential parts, which is important for, e.g., surgery planning.

Part ii started with an overview of the most common feature lines. For the first time, all feature line methods were compared. Therefore, different sets of criteria, such as the possibility to illustrate sharp edges, were analyzed. As a consequence, recommendations for the use of specific feature lines in certain medical applications were derived. Furthermore, an evaluation of feature lines on medical surfaces was conducted. As a result, some requirements and conclusion were drawn. Based on the results of the evaluation, two findings were identified. One result was that feature lines are not able to give a spatial impression of the surface. The other result was that suggestive contours provide the most reasonable result for a variety of anatomical surface models, but additional surface shading is necessary to convey surface information. This thesis presented novel illustrative visual-

ization techniques that overcome the limitations of conventional line drawing techniques.

Additional investigations revealed limitations in hatching methods. Therefore, the ConFIS method was developed to fill the gap between feature line and hatching methods. To strengthen the advantages of ConFIS, two evaluations were conducted to compare other state-of-the-art line drawing techniques with ConFIS. Then, the ConFIS method was improved to a LIC-based approach. In an evaluation, it was confirmed that the LIC-based approach can keep up with the ConFIS method. Furthermore, the LIC-based approach provides a frame-coherent illustration and is more tessellation-independent in comparison to ConFIS. Importantly, the LIC-based method is also able to illustrate animated surfaces in real-time by staying frame-coherent.

Part iii covered the second finding of the evaluation where the suggestive contour method is the most reasonable result, but surface shading is necessary for a spatial impression. Thus, instead of illustrating suggestive contours with surface shading, a combined novel visualization technique is provided. This technique enhanced the most important structures based on the suggestive contours. Furthermore, additional information, such as embedded blood flow, could also be well perceived. The importance and the advantages against other methods were confirmed in an evaluation.

Furthermore, this part covered an illustrative visualization technique that is used for focus-and-context visualization. For the first time, the wall thickness and additional hemodynamic information were given in a surface model for aneurysms. This thesis presented an illustrative visualization technique that depicts different layers of information in a single scene. The importance of this approach was confirmed in an evaluation, too.

9.2 FUTURE WORK

The work presented in this thesis can be extended in different ways. In the following, all parts all listed and several ideas will be presented where the developed algorithms can be improved.

Part i

In this part, a curve smoothing algorithm was presented. Here, the algorithm worked such that the curve points are consecutively relocated. To fasten this process, this algorithm may be executed on the GPU such that each point can be individually treated and relocated at the same time. This would speed up the algorithm and makes it independent of the curve's starting direction. Another idea is to examine if there is an initial scalar field on the surface mesh such that the zero-crossings is defined as the initial curve. If such a scalar field

exists, one could investigate if the smoothing of the scalar field and therefore, the translation of the zero-crossing has a similar result regarding the curve smoothing process.

Part ii

In this part, two different approaches were introduced. ConFIS illustrates only salient regions, which fulfill the conditions mentioned in Subsection 5.2.2. As a consequence of these conditions, convex regions cannot be illustrated. The parameters can be modified in order to emphasize sharp features of the models. However, more parameters are required to define a lower and upper bound. Therefore, the Hessian matrix of the mean curvature scalar field can be determined to obtain the minima and maxima of the surface mesh. The concept of the *contour margin* is used to provide a frame-coherent illustration. An aspect would be to parameterize the contour of the surface model in object space. First, the advantage is to uniformly distribute the seed points for the streamlines equidistant to each other. This provides on the one hand the possibility to apply ConFIS even on low-tessellated surfaces. On the other hand, the streamlines are consistently drawn during the run-time. This would result in a frame-coherent approach, which is independent on the tessellation of the surface model. Another extension is the simulation of light. As ConFIS seeds streamlines at the contour and at the *contour margin*, this conveys the feeling of a headlight. Therefore, the middle parts of cylindrical shapes are illuminated. Following the conclusions of Šoltészová et al. [203], a displacement of the view vector can be considered to simulate different illumination styles. Furthermore, the evaluation can be extended according to Kim et al. [106] and Blair and House [8]. In their experiments, participants had to orient vectors on the model surface manually to fit the perceived surface normal.

Regarding the LIC-based approach, there are also some ideas to extend the method. One aspect is to transfer this method to topology-varying shape models Thus, it can be used for surfaces with varying number of triangles and vertices. Another idea is to change the noise in such a way that only a few lines are drawn. In the current state of the method, noise and LIC generate very dense lines, which may seem a bit like a shading. Especially if the resulting images are too small, the difference is hard to perceive. Regarding the PEL approach by Xie et al. [209], the authors used different lights to filter out noise feature lines. Therefore, the influence on light can be used to vary the result. This can also be used as an interesting parameter for the final result. First, additional lights can be used for shading of the initial result (with a headlight) and second, as a tool for avoiding visual clutter. Xie et al. (PEL) used additional lights for clutter avoidance. Given a region with a lot of (unnecessary) feature lines, additional spotlights can be added which spot on this specific region. Thus, small features

will disappear. Focusing on different light positions may be an interesting influence on the result. Furthermore, attenuating the LIC according to the light position might also lead to pleasant results.

Part iii

For future work, a controlled user study can be considered to quantify the spatial impression of the presented approach in Chapter 7 compared to a semitransparent vessel visualization and the ghosted view approach by Gasteiger et al. [66]. This study is inspired by Baer et al. [6] and includes task-based experiments such as adjusting of surface normals or distance estimations of vessel sections. The quantitative performance of each shading technique can be assessed based on the accuracy and response time of each task. Furthermore, different color bars can be used to decode the distances of pathline-nearby surface regions. This color-coding would improve the assessment of the distances of the pathlines to the vessel surface. Moreover, the distance calculation based on spatial data structures, as described in Zhou et al. [214], can be implemented. They incorporate the k-d tree calculation on the GPU and their method significantly improves of the calculation time.

For the illustration of the wall thickness and hemodynamic information, additional information such as blood flow may be an important aspect for the analysis of aneurysms. The visual exploration of the flow information, however, is a challenging task. Domain experts have to investigate the flow information in combination with its enclosed vessel anatomy because both information strongly correlate. Therefore, applying illustrative visualization techniques for the wall representation with thickness and blood flow information is a crucial task for a complete representation of these data.

Although much work remains to be done, this dissertation showed different illustrative visualization techniques with various advantages. Novel methods that fill a gap in the literature were presented and all the advantages were confirmed in evaluations.

BIBLIOGRAPHY

[1] Rolf Adams and Leanne Bischof. Seeded region growing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(6): 641–647, 1994.

[2] Pierre Alliez, David Cohen-Steiner, Olivier Devillers, Bruno Lévy, and Mathieu Desbrun. Anisotropic polygonal remeshing. In *Proc. of ACM SIGGRAPH*, pages 485–493, 2003.

[3] S. Appanaboyina, F. Mut, R. Löhner, CM Putman, and JR Cebral. Computational Fluid Dynamics of Stented Intracranial Aneurysms Using Adaptive Embedded Unstructured Grids. *Numerical Methods in Fluids*, 57(5):475–493, 2008.

[4] Marco Attene, Sagi Katz, Michela Mortara, Giuseppe Patané, Michela Spagnuolo, and Ayellet Tal. Mesh segmentation - a comparative study. In *IEEE Conf. on Shape Modeling and Applications*, pages 7–18, 2006.

[5] Ragnar Bade, Jens Haase, and Bernhard Preim. Comparison of Fundamental Mesh Smoothing Algorithms for Medical Surface Models. In *Simulation und Visualisierung*, pages 289–304, 2006.

[6] Alexandra Baer, Rocco Gasteiger, Douglas W. Cunningham, and Bernhard Preim. Perceptual Evaluation of Ghosted View Techniques for the Exploration of Vascular Structures and Embedded Flow. *Computer Graphics Forum*, 30(3):811–820, 2011.

[7] Merih I. Baharoglu, Clemens M. Schirmer, Daniel A. Hoit, Bu-Lang Gao, and Adel M. Malek. Aneurysm Inflow-Angle as a Discriminant for Rupture in Sidewall Cerebral Aneurysms Morphometric and Computational Fluid Dynamic Analysis. *Stroke*, 41(7):1423–1430, 2010.

[8] Alethea Bair and Donald H. House. Grid with a view: Optimal texturing for perception of layered surface shape. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1656–1663, 2007.

[9] Lorenzo Baldacci, Matteo Golfarelli, Alessandra Lumini, and Stefano Rizzi. Clustering techniques for protein surfaces. *Pattern recognition*, 39(12):2370–2382, 2006.

[10] Simone Balocco, Carlo Gatta, Francesco Ciompi, and et al. Standardized evaluation methodology and reference database for evaluating IVUS image segmentation. *Computerized medical imaging and graphics*, 38(2):70–90, 2014.

[11] D Balzani, D Böse, D Brands, R Erbel, A Klawonn, O Rheinbach, and J Schröder. Parallel simulation of patient-specific atherosclerotic arteries for the enhancement of intravascular ultrasound diagnostics. *Engineering Computations*, 29(8):888–906, 2012.

[12] Mikhail Belkin, Jian Sun, and Yusu Wang. Discrete laplace operator on meshed surfaces. In *Proc. of Symposium on Computational Geometry*, pages 278–287. ACM, 2008.

[13] Halim Benhabiles, Guillaume Lavoué, Jean Phillipe Vandeborre, and Mohamed Daoudi. Learning boundary edges for 3D-mesh segmentation. *Computer Graphics Forum*, 30(8):2170–2182, 2011.

[14] Jon Louis Bentley. Multidimensional Binary Search Trees Used for Associative Searching. *Communications ACM*, 18(9):509–517, 1975.

[15] Helen M. Berman, John Westbrook, Zukang Feng, Gary Gilliland, T. N. Bhat, Helge Weissig, Ilya N. Shindyalov, and Philip E. Bourne. The protein data bank. *Nucleic Acids Res*, 28: 235–242, 2000. URL `http://www.pdb.org`.

[16] Stephan Bischoff, Tobias Weyand, and Leif Kobbelt. Snakes on triangle meshes. In *Proc. of Bildverarbeitung für die Medizin*, pages 208–212, 2005.

[17] David Bommes and Leif Kobbelt. Accurate computation of geodesic distance fields for polygonal curves on triangle meshes. In *Proc. of Vision, Modeling and Visualization*, pages 151–160, 2007.

[18] S. Born, M. Pfeifle, M. Markl, M. Gutberlet, and G. Scheuermann. Visual Analysis of Cardiac 4D MRI Blood Flow Using Line Predicates. *IEEE Transactions on Visualization and Computer Graphics*, 19(6):900–912, 2013.

[19] Silvia Born, Alexander Wiebel, Jan Friedrich, Gerik Scheuermann, and Dirk Bartz. Illustrative stream surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1329–1338, 2010.

[20] Silvia Born, Michael Markl, Matthias Gutberlet, and G. Scheuermann. Illustrative Visualization of Cardiac and Aortic Blood Flow from 4D MRI Data. In *IEEE Pacific Visualization*, pages 129 –136, 2013.

[21] Mario Botsch, Leif Kobbelt, Mark Pauly, Pierre Alliez, and Br uno Levy. *Polygon Mesh Processing*. AK Peters, 2010.

[22] Andrea Brambilla, Robert Carnecky, Ronald Peikert, Ivan Viola, and Helwig Hauser. Illustrative flow visualization: State of the art, trends and challenges. In *EuroGraphics State of the Art Reports (STARs)*, pages 75–94, 2012.

[23] Stefan Bruckner. *Interactive Illustrative Volume Visualization*. PhD thesis, Institute of Computer Graphics and Algorithms, Vienna University of Technology, 3 2008.

[24] Stefan Bruckner and Meister Eduard Gröller. Volumeshop: An interactive system for direct volume illustration. In *IEEE Transactions on Visualization and Computer Graphics*, pages 671–678, 2005.

[25] Stefan Bruckner and Meister Eduard Gröller. Exploded views for volume data. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1077–1084, 9 2006.

[26] Stefan Bruckner, Sören Grimm, Armin Kanitsar, and Meister Eduard Gröller. Illustrative context-preserving exploration of volume data. *IEEE Transactions on Visualization and Computer Graphics*, 12(6):1559–1569, 11 2006.

[27] Stefan Bruckner, Eduard Gröller, Klaus Mueller, Bernhard Preim, and Deborah Silver. Illustrative focus+context approaches in interactive volume visualization. In *Scientific Visualization: Advanced Concepts*, chapter 10. 2010.

[28] Stefan Bruckner, Peter Rautek, Ivan Viola, Mike Roberts, Mario Costa Sousa, and M. Eduard Gröller. Hybrid visibility compositing and masking for illustrative rendering. *Computers & Graphics*, 34(4):361 – 369, 2010. Procedural Methods in Computer Graphics Illustrative Visualization.

[29] Kevin Buchin and Maike Walther. *Hatching, Stroke Styles & Pointillism*, volume ShaderX2 - Shader Tips and Tricks, chapter Rendering Techniques, pages 340–347. Wordware Publishing, 2003.

[30] Michael Burns, Janek Klawe, Szymon Rusinkiewicz, Adam Finkelstein, and Doug DeCarlo. Line drawings from volume data. *Proc. of ACM SIGGRAPH*, 24(3):512–518, 2005.

[31] Brian Cabral and Leith Casey Leedom. Imaging vector fields using line integral convolution. In *Proc. of ACM SIGGRAPH*, pages 263–270, 1993.

[32] F. Cazals and M. Pouget. Estimating differential quantities using polynomial fitting of osculating jets. In *Proc. of ACM SIGGRAPH*, pages 177–187, 2003.

[33] J R Cebral, M Sheridan, and C M Putman. Hemodynamics and bleb formation in intracranial aneurysms. *American Journal of Neuroradiology*, 31(2):304–310, 2010.

[34] Juan R. Cebral, Marcelo Adrián Castro, Sunil Appanaboyina, Christopher M. Putman, Daniel Millan, and Alejandro F. Frangi. Efficient Pipeline for Image-Based Patient-Specific Analysis of Cerebral Aneurysm Hemodynamics: Technique and Sensitivity. *IEEE Transactions on Medical Imaging*, 24(4):457–467, 2005.

[35] Juan R. Cebral, Fernando Mut, J. Weir, and Christopher M. Putman. Association of Hemodynamic Characteristics and Cerebral Aneurysm Rupture. *American Journal of Neuroradiology*, 32 (2):264–270, 2011.

[36] George Chaikin. An algorithm for high speed curve generation. *Computer Graphics and Image Processing*, 3:346–349, 1974.

[37] Matthieu Chavent, Bruno Lévy, Michael Krone, Katrin Bidmon, Jean-Philippe Nominé, Thomas Ertl, and Marc Baaden. GPU-powered tools boost molecular visualization. *Brief. Bioinform.*, 12(6):689–701, 2011.

[38] Xin Chen and Francis Schmitt. Intrinsic surface properties from surface triangulation. In *Proc. of the ECCV*, pages 739–743, 1992.

[39] Alan Chu, Wing-Yin Chan, Jixiang Guo, Wai-Man Pang, and Pheng-Ann Heng. Perception-aware depth cueing for illustrative vascular visualization. *BioMedical Engineering and Informatics, International Conference on*, 1:341–346, 2008.

[40] David Cohen-Steiner and Jean-Marie Morvan. Restricted delaunay triangulations and normal cycle. In *Proc. of SCG*, pages 312–321. ACM, 2003.

[41] Forrester Cole, Aleksey Golovinskiy, Alex Limpaecher, Heather Stoddart Barros, Adam Finkelstein, Thomas Funkhouser, and Szymon Rusinkiewicz. Where do people draw lines? *Proc. of ACM SIGGRAPH*, 27(3), August 2008.

[42] M. L. Connolly. Analytical molecular surface calculation. *Journal of Applied Crystallography*, 16(5):548–558, Oct 1983.

[43] Carlos Correa, Deborah Silver, and Min Chen. Feature aligned volume manipulation for illustration and visualization. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1069–1076, 2006.

[44] Keenan Crane, Clarisse Weischedel, and Max Wardetzky. *Geodesics in Heat*. Proc. of ACM SIGGRAPH, 2013.

[45] Balázs Csebfalvi, Balázs Tóth, Stefan Bruckner, and Meister Eduard Gröller. Illumination-driven opacity modulation for expressive volume rendering. In *Proc. of Vision, Modeling & Visualization*, pages 103–109, November 2012.

[46] Doug DeCarlo, Adam Finkelstein, Szymon Rusinkiewicz, and Anthony Santella. Suggestive contours for conveying shape. *Proc. of ACM SIGGRAPH*, pages 848–855, 2003.

[47] Christian Dick, Rainer Burgkart, and Rüdiger Westermann. Distance visualization for interactive 3D implant planning. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2173–2182, 2011.

[48] Joachim Diepstraten, Daniel Weiskopf, and Thomas Ertl. Interactive Cutaway Illustrations. *Computer Graphics Forum*, 22(3): 523–532, 2003.

[49] Edsger W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[50] Manfredo P. do Carmo. *Differential Geometry of Curves and Surfaces*. Prentice-Hall, Englewood Cliffs, NJ, 1976.

[51] Manfredo P. do Carmo. *Riemannian Geometry*. Birkhäuser, Boston, MA, 1992.

[52] H. Doleisch, M. Gasser, and H. Hauser. Interactive Feature Specification for Focus+Context Visualization of Complex Simulation Data. In *Joint Eurographics / IEEE TCVG Symposium on Visualization*, pages 239–248, 2003.

[53] Feng Dong, Gordon J. Clapworthy, Hai Lin, and Meleagros A. Krokos. Nonphotorealistic rendering of medical volume data. *IEEE Computer Graphics and Applications*, 23(4):44–52, 2003.

[54] Debra Dooley and Michael F. Cohen. Automatic illustration of 3d geometric models: Lines. *Proc. of ACM SIGGRAPH*, 24(2): 77–82, 1990.

[55] B. S. Duncan and A. J. Olson. Texture mapping parametric molecular surfaces. *Journal of Molecular Graphics*, 13(4):258–264, 1995.

[56] Nira Dyn, David Levin, and Dingyuan Liu. Interpolatory convexity-preserving subdivision schemes for curves and surfaces. *Computer-Aided Design*, pages 211–216, 1992.

[57] David Ebert and Penny Rheingans. Volume illustration: Nonphotorealistic rendering of volume models. In *IEEE Transactions on Visualization and Computer Graphics*, pages 195–202, 2000.

[58] Maarten Everts, Henk Bekker, Jos B. T. M. Roerdink, and Tobias Isenberg. Illustrative Line Styles for Flow Visualization. In *Pacific Graphics*, pages 105–110. Eurographics Association, 2011. Short paper.

[59] Maarten H. Everts, Henk Bekker, Jos B. T. M. Roerdink, and Tobias Isenberg. Depth-dependent halos: Illustrative rendering of dense line data. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1299–1306, 2009.

[60] Martin Falk and Daniel Weiskopf. Output-sensitive 3d line integral convolution. *IEEE Transactions on Visualization and Computer Graphics*, 14(4):820–834, 2008.

[61] Martin Falk, Michael Klann, Matthias Reuss, and Thomas Ertl. Visualization of signal transduction processes in the crowded environment of the cell. In *IEEE Pacific Visualization*, pages 169–176, 2009.

[62] Martin Falk, Michael Krone, and Thomas Ertl. Atomistic visualization of mesoscopic whole-cell simulations using ray-casted instancing. *Computer Graphics Forum*, 32(8):195–206, 2013.

[63] J. Fischer, D. Bartz, and W. Straßer. Illustrative Display of Hidden Iso-Surface Structures. In *IEEE Transactions on Visualization and Computer Graphics*, pages 663–670, October 2005.

[64] Michael S. Floater. Mean value coordinates. *CAGD*, 20(1):19 – 27, 2003.

[65] Rocco Gasteiger, Christian Tietjen, Alexandra Baer, and Bernhard Preim. Curvature- and model-based surface hatching of anatomical structures derived from clinical volume datasets. In *Smart Graphics*, pages 255–262, 2008.

[66] Rocco Gasteiger, Mathias Neugebauer, Christoph Kubisch, and Bernhard Preim. Adapted Surface Visualization of Cerebral Aneurysms with Embedded Blood Flow Information. In *EG Workshop on Visual Computing for Biology and Medicine*, pages 25–32, 2010.

[67] Rocco Gasteiger, Mathias Neugebauer, Oliver Beuing, and Bernhard Preim. The FlowLens: A Focus-and-Context Visualization Approach for Exploration of Blood Flow in Cerebral Aneurysms. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2183–2192, 2011.

[68] Weidong Geng. *The Algorithms and Principles of Non-photorealistic Graphics: Artistic Rendering and Cartoon Animation*. Springer, 1 edition, 2010.

[69] M. Gerl and T. Isenberg. Interactive example-based hatching. *Computers & Graphics*, 2013.

[70] Moritz Gerl, Peter Rautek, Tobias Isenberg, and Eduard Gröller. Semantics by analogy for illustrative volume visualization. *Computers & Graphics*, 36(3):201 – 213, 2012.

[71] D. Gil, P. Radeva, J. Saludes, and J. Mauri. Automatic segmentation of artery wall in coronary IVUS images: a probabilistic approach. In *Proc. of Computers in Cardiology*, pages 687–690, 2000.

[72] Ahna Girshick and Victoria Interrante. Real-time principal direction line drawings of arbitrary 3d surfaces. In *Proc. of ACM SIGGRAPH*, page 271, 1999.

[73] Sylvia Glaßer, Steffen Oeltze, Anja Hennemuth, Christoph Kubisch, A. Mahnken, S. Wilhelmsen, and Bernhard Preim. Automatic Transfer Function Specification for Visual Emphasis of Coronary Artery Plaque. *Computer Graphics Forum*, 29(1):191–201, 2010.

[74] Jack Goldfeather and Victoria Interrante. A novel cubic-order algorithm for approximating principal direction vectors. *Proc. of ACM SIGGRAPH*, 23(1):45–63, 2004.

[75] Bruce Gooch and Amy Gooch. *Non-Photorealistic Rendering*. A.K. Peters, 2001.

[76] D S Goodsell and A J Olson. Molecular illustration in black and white. *J Mol Graph*, 10:235–240, 1992.

[77] D.S. Goodsell. *The Machinery of Life*. Biomedical and Life Sciences. Copernicus Books, 2009.

[78] Martin Haidacher, Daniel Patel, Stefan Bruckner, Armin Kanitsar, and Meister Eduard Gröller. Volume visualization based on statistical transfer-function spaces. In *IEEE Pacific Visualization*, pages 17–24, March 2010.

[79] E. Hameiri and I. Shimshoni. Estimating the principal curvatures and the darboux frame from real 3-d range data. *Trans. Sys. Man Cyber. Part B*, 33(4):626–637, 2003.

[80] C. Hansen, J. Wieferich, F. Ritter, C. Rieder, and H.-O. Peitgen. Illustrative visualization of 3d planning models for augmented reality in liver surgery. *CARS*, 5(2):133–141, 2010.

[81] Philip Hartman. *Ordinary Differential Equations*. John Wiley & Sons, 1964.

[82] Anja Hennemuth, Ola Friman, Christian Schumann, Jelena Bock, Johann Drexl, Michael Markl, and Heinz-Otto Peitgen. Fast Interactive Exploration of 4D MRI Flow Data. In *Proc. SPIE Medical Imaging*, 2011.

[83] Aaron Hertzmann and Denis Zorin. Illustrating smooth surfaces. In *Proc. of ACM SIGGRAPH*, pages 517–526, 2000.

[84] Fernando Vega Higuera, Natascha Sauber, Bernd Tomandl, Christopher Nimsky, Guenther Greiner, and Peter Hastreiter. Automatic adjustment of bidimensional transfer functions for direct volume visualization of intracranial aneurysms. In *Proc. of SPIE Medical Imaging*, pages 275–284, 2004.

[85] Michael Hofer and Helmut Pottmann. Energy-minimizing splines in manifolds. In *Proc. of ACM SIGGRAPH*, pages 284–293, 2004.

[86] MichaelD. Hope, S.Jarrett Wrenn, and Petter Dyverfeldt. Clinical applications of aortic 4d flow imaging. *Current Cardiovascular Imaging Reports*, 6(2):128–139, 2013.

[87] Josef Hoschek and Dieter Lasser. *Fundamentals of Computer Aided Geometric Design*. AK Peters, 1993.

[88] Jin Huang, Wenjie Pei, Chunfeng Wen, Guoning Chen, Wei Chen, and Hujun Bao. Output-coherent image-space lic for surface flow visualization. *IEEE Pacific Visualization*, pages 137–144, 2012.

[89] Mathias Hummel, Christoph Garth, Bernd Hamann, Hans Hagen, and Kenneth I. Joy. Iris: Illustrative Rendering for Integral Surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1319–1328, 2010.

[90] William Humphrey, Andrew Dalke, and Klaus Schulten. VMD – Visual Molecular Dynamics. *J Mol Graph*, 14:33–38, 1996.

[91] Victoria Interrante, Henry Fuchs, and Stephen Pizer. Enhancing transparent skin surfaces with ridge and valley lines. In *IEEE Transactions on Visualization and Computer Graphics*, pages 52–59, 1995.

[92] Victoria Interrante, Henry Fuchs, and Stephen M. Pizer. Conveying the 3D Shape of Smoothly Curving Transparent Surfaces via Texture. *IEEE Transactions on Visualization and Computer Graphics*, 3(2):98–117, 1997.

[93] Tobias Isenberg, Bert Freudenberg, Nick Halper, Stefan Schlechtweg, and Thomas Strothotte. A developer's guide to silhouette algorithms for polygonal models. *IEEE Computer Graphics and Applications*, 23(4):28–37, July 2003.

[94] Tobias Isenberg, Petra Neumann, Sheelagh Carpendale, Mario Costa Sousa, and Joaquim A. Jorge. Non-photorealistic rendering in context: An observational study. In *Proc. of NPAR*, pages 115–126. ACM, 2006.

[95] Fujimaro Ishida, Hiroyuki Ogawa, Takeo Simizu, Tadashi Kojima, and Waro Taki. Visualizing the dynamics of cerebral aneurysms with four-dimensional computed tomographic angiography. *Neurosurgery*, 57(3):460–471, 2005.

[96] Werner M. Jainek, Silvia Born, Dirk Bartz, Wolfgang Straßer, and Jan Fischer. Illustrative hybrid visualization and exploration of anatomical and functional brain data. *Computer Graphics Forum*, 27(3):855–862, 2008.

[97] Ik-Kyung Jang, Brett E Bouma, Dong-Heon Kang, Seung-Jung Park, Seong-Wook Park, Ki-Bae Seung, Kyu-Bo Choi, Milen Shishkov, Kelly Schlendorf, Eugene Pomerantsev, et al. Visualization of coronary atherosclerotic plaques in patients using optical coherence tomography: comparison with intravascular ultrasound. *Journal of the American College of Cardiology*, 39(4): 604–609, 2002.

[98] Zhongping Ji, Ligang Liu, Zhonggui Chen, and Guojin Wang. Easy mesh cutting. *Computer Graphics Forum*, 25(3):283–291, 2006.

[99] Tilke Judd, Frédo Durand, and Edward Adelson. Apparent ridges for line drawing. In *Proc. of ACM SIGGRAPH*, page 19, 2007.

[100] Moonryul Jung and Haengkang Kim. Snaking across 3d meshes. In *Proceedings Pacific Graphics*, pages 87–93, 2004.

[101] Seppo Juvela. Prevalence of and Risk Factors for Intracranial Aneurysms. *The Lancet Neurology*, 10(7):595–597, 2011.

[102] Vladimir Kajalin. Screen-Space Ambient Occlusion. In *ShaderX*^{7}, pages 413–424. Charles River Media, 2009.

[103] Lotan Kaplansky and Ayellet Tal. Mesh segmentation refinement. *Computer Graphics Forum*, 28(7):1995–2003, 2009.

[104] C. Karmonik, J. Bismuth, MG Davies, and AB Lumsden. Computational Fluid Dynamics as a Tool for Visualizing Hemodynamic Flow Patterns. *Methodist DeBakey Cardiovascular Journal*, 5(3):26–33, 2009.

[105] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, pages 321–331, 1988.

[106] Sunghee Kim, Haleh Hagh-Shenas, and Victoria Interrante. Conveying Shape with Texture: Experimental Investigations of Texture's Effects on Shape Categorization Judgments. *IEEE Transactions on Visualization and Computer Graphics*, 10(4):471–483, 2004.

[107] Ron Kimmel and James A. Sethian. Fast marching methods for computing distance maps shortest paths. *CPAM Report 669, University of California, Berkeley*, 1996.

[108] Gordon Kindlmann, Ross Whitaker, Tolga Tasdizen, and Torsten Moller. Curvature-based transfer functions for direct volume rendering: Methods and applications. In *IEEE Transactions on Visualization and Computer Graphics*, VIS '03, pages 67–. IEEE Computer Society, 2003.

[109] Jan J. Koendrink, Andrea J. van Doorn, and Astrid M. L. Kappers. Surface perception in pictures. *Perception & Psychophysics*, 52(5):487–496, 1992.

[110] B. Köhler, R. Gasteiger, U. Preim, H. Theisel, M. Gutberlet, and B. Preim. Semi-Automatic Vortex Extraction in 4D PC-MRI Cardiac Blood Flow Data using Line Predicates. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2773–2782, 2013.

[111] Michael Kolomenkin, Ilan Shimshoni, and Ayellet Tal. Demarcating curves for shape illustration. In *Proc. of ACM SIGGRAPH Asia*, pages 157:1–157:9, 2008.

[112] Michael Krone, Katrin Bidmon, and Thomas Ertl. Interactive visualization of molecular surface dynamics. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1391–1398, 2009.

[113] Michael Krone, John E. Stone, Thomas Ertl, and Klaus Schulten. Fast visualization of gaussian density surfaces for molecular dynamics and particle system trajectories. In *EG EuroVis Short Papers*, volume 1, pages 67–71, 2012.

[114] Arno Krüger, Christian Tietjen, Jana Hintze, Bernhard Preim, Ilka Hertel, and Gero Strauß. Analysis and exploration of 3d visualization for neck dissection planning. *CARS*, 1281(0):497 – 503, 2005.

[115] Arno Krüger, Christoph Kubisch, Gero Strauß, and Bernhard Preim. Sinus Endoscopy - Application of Advanced GPU Volume Rendering for Virtual Endoscopy. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1491–1498, 2008.

[116] Jens Krüger, Jens Schneider, and Rüdiger Westermann. Clearview: An interactive context preserving hotspot visualiza-

tion technique. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):941–948, 2006.

[117] W. Kühnel. *Differential Geometry: Curves - Surfaces - Manifolds*. Student mathematical library. American Mathematical Society, 2006. ISBN 9780821839881.

[118] Yu-Kun Lai, Qian-Yi Zhou, Shi-Min Hu, Johannes Wallner, and Helmut Pottmann. Robust feature classification and editing. *IEEE Transactions on Visualization and Computer Graphics*, pages 34–45, 2007.

[119] Ove Daae Lampe, Ivan Viola, Nathalie Reuter, and Helwig Hauser. Two-level approach to efficient visualization of protein dynamics. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1616–1623, 2007.

[120] David A. Lane. Visualization of time-dependent flow fields. In Gregory M. Nielson and R. Daniel Bergeron, editors, *IEEE Transactions on Visualization and Computer Graphics*, pages 32–38. IEEE Computer Society, 1993.

[121] David A. Lane. Scientific visualization of large-scale unsteady fluid flows. In Gregory M. Nielson, Hans Hagen, and Heinrich Müller, editors, *Scientific Visualization*, pages 125–145. IEEE Computer Society, 1994.

[122] RobertS. Laramee, Helwig Hauser, Lingxiao Zhao, and FritsH. Post. Topology-Based Flow Visualization, The State of the Art. In Helwig Hauser, Hans Hagen, and Holger Theisel, editors, *Topology-based Methods in Visualization*, Mathematics and Visualization, pages 1–19. Springer Berlin Heidelberg, 2007. ISBN 978-3-540-70822-3.

[123] Yunjin Lee and Seungyong Lee. Geometric snakes for triangular meshes. *Computer Graphics Forum*, pages 229–238, 2002.

[124] Yunjin Lee, Seungyong Lee, Ariel Shamir, Daniel Cohen-Or, and Hans-Peter Seidel. Intelligent mesh scissoring using 3d snakes. In *Proceedings Pacific Graphics*, pages 279–287, 2004.

[125] David Lesage, Elsa D. Angelini, Isabelle Bloch, and Gareth Funka-Lea. A Review of 3D Vessel Lumen Segmentation Techniques: Models, Features and Extraction Schemes. *Medical Image Analysis*, 13(6):819–845, 2009.

[126] Bruno Lévy, Sylvain Petitjean, Nicolas Ray, and Jérome Maillot. Least squares conformal maps for automatic texture atlas generation. *Proc. of ACM SIGGRAPH*, 21(3):362–371, July 2002.

[127] Rensis Likert. A Technique for the Measurement of Attitudes. *Archives of Psychology*, 140:1–55, 1932.

[128] Dan R. Lipsa, Robert S. Laramee, Simon J. Cox, Jonathan C. Roberts, Rick Walker, Michelle Borkin, and Hanspeter Pfister. Visualization for the physical sciences. *Computer Graphics Forum*, 31(8):2317–2347, 2012.

[129] Nan Liu and Ming-Yong Pang. A survey of shadow rendering algorithms: Projection shadows and shadow volumes. In *Computer Science and Engineering*, volume 1, pages 488–492, 2009.

[130] Thomas Luft, Carsten Colditz, and Oliver Deussen. Image enhancement by unsharp masking the depth buffer. *Proc. of ACM SIGGRAPH*, 25(3):1206–1213, 2006.

[131] P. J. Kilner M. Markl and T. Ebbers. Comprehensive 4D velocity mapping of the heart and great vessels by cardiovascular magnetic resonance. *Journal of Cardiovascular Magnetic Resonance*, 13: 7, 2011.

[132] Li Ma and Dezhong Chen. Curve shortening in a riemannian manifold. *Annali di Matematica Pura ed Applicata*, 186(4):663–684, 2007.

[133] R. MacNeal. *The Solution of Partial Differential Equations by Means of Electrical Networks*. PhD thesis. California Institute of Technology, 1949.

[134] Michael Markl, Alex Frydrychowicz, Sebastian Kozerke, Mike Hope, and Oliver Wieben. 4D flow MRI. *Journal of Magnetic Resonance Imaging*, 36(5):1015–1036, 2012.

[135] D. Marr. Early Processing of Visual Information. *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences*, 275(942):483–519, 1976.

[136] Dimas Martínez, Luiz Velho, and Paulo C. Carvalho. Computing geodesics on triangular meshes. *Computers & Graphics*, 29 (5):667–675, 2005.

[137] Oliver Mattausch, Thomas Theußl, Helwig Hauser, and Eduard Gröller. Strategies for Interactive Exploration of 3D Flow Using Evenly-Spaced Illuminated Streamlines. In *ACM Spring Conference on Computer Graphics*, pages 230–241, 2003.

[138] Nelson Max. Weights for computing vertex normals from facet normals. *J. Graph. Tools*, 4(2):1–6, 1999.

[139] Tony McLoughlin, Robert S. Laramee, Ronald Peikert, Frits H. Post, and Min Chen. Over Two Decades of Integration-Based,

Geometric Flow Visualization. *Computer Graphics Forum*, 29(6): 1807–1829, 2010.

[140] Mark Meyer, Mathieu Desbrun, Peter Schröder, and Alan H. Barr. Discrete differential-geometry operators for triangulated 2-manifolds. In *Proc. VisMath*, pages 35–57, 2002.

[141] David Meyers, Shelley Skinner, and Kenneth Sloan. Surfaces from contours. *Proc. of ACM SIGGRAPH*, 11(3):228–258, 1992.

[142] Joseph S. B. Mitchell, David M. Mount, and Christos H. Papadimitriou. The discrete geodesic problem. *SIAM J. Comput.*, 16(4):647–668, August 1987.

[143] Tobias Mönch, Rocco Gasteiger, Gabor Janiga, Holger Theisel, and Bernhard Preim. Context-Aware Mesh Smoothing for Biomedical Applications. *Computers & Graphics*, 35(4)(4):755 – 767, 2011.

[144] Tobias Mönch, Mathias Neugebauer, and Bernhard Preim. Optimization of Vascular Surface Models for Computational Fluid Dynamics and Rapid Prototyping. In *Second International Workshop on Digital Engineering*, pages 16–23, 2011.

[145] Jorge J. Moré. The Levenberg-Marquardt algorithm: Implementation and theory. In G. A. Watson, editor, *Numerical Analysis*, pages 105–116. Springer, 1977.

[146] Dimas Martínez Morera, Paulo Cezar Carvalho, and Luiz Velho. Geodesic bezier curves: a tool for modeling on triangulations. In *Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI)*, pages 71–78, 2007.

[147] Dimas Martínez Morera, Luiz Velho, and Paulo Cezar Carvalho. Subdivision curves on triangular meshes. *Proceedings of 13th Iberoamerican Congress on Pattern Recognition (CIARP)*, 2008.

[148] R. Muthukrishnan and M. Radha. Edge detection techniques for image segmentation. *IJCSIT*, 3(6), 2011.

[149] Ehsan Nadernejad, Sara Sharifzadeh, and Hamid Hassanpour. Edge detection techniques: Evaluations and comparisons. *Applied Mathematical Sciences*, 2(no. 31):1507 – 1520, 2008.

[150] Mathias Neugebauer, Rocco Gasteiger, Oliver Beuing, Volker Diehl, Martin Skalej, and Bernhard Preim. Map Displays for the Analysis of Scalar Data on Cerebral Aneurysm Surfaces. *Computer Graphics Forum*, 28(3):895–902, 2009.

[151] Alex Ni, Kyuman Jeong, Seungyong Lee, and Lee Markosian. Multi-scale line drawings from 3d meshes. In *Proc. of Interactive 3D graphics and games*, pages 133–137, 2006.

[152] Gregory M. Nielson and Il-Hong Jung. Tools for computing tangent curves for linearly varying vector fields over tetrahedral domains. *IEEE Transactions on Visualization and Computer Graphics*, 5(4):360–372, October 1999.

[153] Seán I. O'Donoghue, David S. Goodsell, Achilleas S. Frangakis, Fabrice Jossinet, Roman A. Laskowski, Michael Nilges, Helen R. Saibil, Andrea Schafferhans, Rebecca C. Wade, Eric Westhof, and Arthur J. Olson. Visualization of macromolecular structures. *Nature methods*, 7(3 Suppl), 2010.

[154] Steffen Oeltze, Anja Hennemuth, Sylvia Glaßer, Caroline Kühnel, and Bernhard Preim. Glyph-Based Visualization of Myocardial Perfusion Data and Enhancement with Contractility and Viability Information. In *Proc. of VCBM*, pages 11–20, 2008.

[155] Yutaka Ohtake, Alexander Belyaev, and Hans-Peter Seidel. Ridge-valley lines on meshes via implicit surface fitting. *Proc. of ACM SIGGRAPH*, 23:609–612, 2004.

[156] D. L. Page, A. Koschan, Y. Sun, J. Paik, and M. A. Abidi. Robust crease detection and curvature estimation of piecewise smooth surfaces from triangle mesh approximations using normal voting. In *CVPR*, pages 162–167, 2001.

[157] James Morris Page. *Ordinary Differential Equations - An Elementary Text-Book With An Introduction To Lies s Theory Of The Group Of One Parameter*. MacMillian and CO., 1897.

[158] Julius Parulek, Timo Ropinski, and Ivan Viola. Importance driven visualization of molecular surfaces, October 2013. IEEE Symposium on Biological Data Visualization.

[159] Roy F. P. Van Pelt. *Real-time Illustrative Visualization of Cardiovascular Hemodynamics*. PhD thesis, Technical University of Eindhoven, 2012.

[160] Michel Petitjean. On the analytical calculation of van der waals surfaces and volumes: Some numerical aspects. *J. Comput. Chem.*, 15(5):507–523, May 1994.

[161] Konrad Polthier and Markus Schmies. Straightest geodesics on polyhedral surfaces. In *SIGGRAPH courses*, pages 30–38, 2006.

[162] Helmut Pottmann and Michael Hofer. A variational approach to spline curves on surfaces. *Comput. Aided Geom. Des.*, 22(7): 693–709, 2005.

[163] Emil Praun, Hugues Hoppe, Matthew Webb, and Adam Finkelstein. Real-time hatching. In *Proc. of ACM SIGGRAPH*, pages 579–584, 2001.

[164] Bernhard Preim and Charl Botha. *Visual Computing for Medicine, 2nd Edition*. 2013.

[165] Peter Rautek, Stefan Bruckner, Eduard Gröller, and Ivan Viola. Illustrative visualization: New technology or useless tautology? *Proc. of ACM SIGGRAPH*, 42(3):4:1–4:8, 2008.

[166] F. M. Richards. Areas, volumes, packing, and protein structure. *Annu. Rev Biophys. Bio.*, 6(1):151–176, 1977.

[167] Timothy J. Richmond. Solvent accessible surface area and excluded volume in proteins: Analytical equations for overlapping spheres and implications for the hydrophobic effect. *Journal of Molecular Biology*, 178(1):63 – 89, 1984.

[168] Felix Ritter, Christian Hansen, Volker Dicken, Olaf Konrad, Bernhard Preim, and Heinz-Otto Peitgen. Real-time illustration of vascular structures. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):877–884, 2006.

[169] Szymon Rusinkiewicz. Estimating curvatures and their derivatives on triangle meshes. In *Symposium on 3D Data Processing, Visualization, and Transmission*, September 2004.

[170] Szymon Rusinkiewicz, Forrester Cole, Doug DeCarlo, and Adam Finkelstein. Line drawings from 3d models. In *Proc. of ACM SIGGRAPH*, pages 39:1–39:356, July 2008.

[171] Takafumi Saito and Tokiichiro Takahashi. Comprehensible rendering of 3-d shapes. *Proc. of ACM SIGGRAPH*, 24(4):197–206, 1990.

[172] G. Sakas, L. Schreyer, and M. Grimm. Preprocessing and volume rendering of 3d ultrasonic data. *IEEE Computer Graphics and Applications*, 15(4):47–54, 1995.

[173] Michael P. Salisbury, Sean E. Anderson, Ronen Barzel, and David H. Salesin. Interactive pen-and-ink illustration. In *Proc. of Computer Graphics and Interactive Techniques*, SIGGRAPH, pages 101–108. ACM, 1994.

[174] Michael P. Salisbury, Michael T. Wong, John F. Hughes, and David H. Salesin. Orientable textures for image-based pen-and-ink illustration. In *Proc. of Computer Graphics and Interactive Techniques*, SIGGRAPH, pages 401–406, 1997.

[175] Tobias Salzbrunn, Heike Jänicke, Thomas Wischgoll, and Gerik Scheuermann. The State of the Art in Flow Visualization: Partition-Based Techniques. In *Simulation and Visualization*, pages 75–92, 2008.

[176] Michel F. Sanner, Arthur J. Olson, and Jean-Claude Spehner. Reduced Surface: An efficient way to compute molecular surfaces. *Biopolymers*, 38(3):305–320, 1996.

[177] Christophe Schlick. A customizable reflectance model for everyday rendering. In *Proc. of Eurographics Workshop on Rendering*, pages 73–83, 1993.

[178] Joachim Schöberl. NETGEN: An Advancing Front 2D/3D-Mesh Generator Based on Abstract Rules. *Computing and Visualization in Science*, 1:41–52, 1997.

[179] N. Senthilkumaran and R. Rajesh. Edge detection techniques for image segmentation – a survey of soft computing approaches. *International Journal of Recent Trends in Engineering*, 1(2), May 2009.

[180] Ariel Shamir. A survey on mesh segmentation techniques. *Computer Graphics Forum*, 27(6):1539–1556, 2008.

[181] Christian Sigg, Tim Weyrich, Mario Botsch, and Markus Gross. GPU-based ray-casting of quadratic surfaces. In *Eurographics Symposium on Point-Based Graphics*, pages 59–65, 2006.

[182] Peter Sikachev, Peter Rautek, Stefan Bruckner, and Meister Eduard Gröller. Dynamic focus+context for volume rendering. In *Proc. of Vision, Modeling and Visualization*, pages 331–338, November 2010.

[183] Olga Sorkine. Laplacian Mesh Processing. pages 53–70, Dublin, Ireland, September 2005. Eurographics Association. Eurographics 05 STAR.

[184] Detlev Stalling and Hans-Christian Hege. Fast and resolution independent line integral convolution. In *Proc. of ACM SIGGRAPH*, pages 249–256, 1995.

[185] Thomas Strothotte and Stefan Schlechtweg. *Non-photorealistic Computer Graphics: Modeling, Rendering, and Animation*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2002. ISBN 1-55860-787-0.

[186] Thomas Strothotte, Bernhard Preim, Andreas Raab, Jutta Schumann, and David R. Forsey. How to Render Frames and Influence People. *Computer Graphics Forum*, 13(3):455–466, 1994.

[187] Vitaly Surazhsky and Tatiana Surazhsky. Fast exact and approximate geodesics on meshes. *Proc. of ACM SIGGRAPH*, 24: 553–560, 2005.

[188] Nikolai A Svakhine, David S Ebert, and William M Andrews. Illustration-Inspired Depth Enhanced Volumetric Medical Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 15(1):77–86, 2009.

[189] Pjotr Svetachov, Maarten H. Everts, and Tobias Isenberg. DTI in context: Illustrating brain fiber tracts in situ. *Computer Graphics Forum*, 29(3):1023–1032, 2010.

[190] T. Taerum, M. C. Sousa, F. Samavati, S. Chan, and J. R. Mitchell. Real-time super resolution contextual close-up of clinical volumetric data. In *IEEE VGTC Conference on Visualization*, pages 347–354, 2006.

[191] Marco Tarini, Paolo Cignoni, and Claudio Montani. Ambient occlusion and edge cueing for enhancing real time molecular visualization. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1237–1244, 2006.

[192] G. Taubin. Estimating the tensor of curvature of a surface from a polyhedral approximation. In *Proc. of ICCV*, pages 902–. IEEE Computer Society, 1995.

[193] Gabriel Taubin. A signal processing approach to fair surface design. In *Proc. of ACM SIGGRAPH*, pages 351–358, 1995.

[194] Christian Teitzel, Roberto Grosso, and Thomas Ertl. Efficient and Reliable Integration Methods for Particle Tracing in Unsteady Flows on Discrete Meshes. In *Visualization in Scientific Computing*, pages 31–41, 1997.

[195] Christian Tietjen, Tobias Isenberg, and Bernhard Preim. Combining silhouettes, surface, and volume rendering for surgery education and planning. In *Computer Graphics Forum*, pages 303–310, 2005.

[196] Christian Tietjen, Roland Pfisterer, Alexandra Baer, Rocco Gasteiger, and Bernhard Preim. Hardware-Acceelerated Illustrative Medical Surface Visualization with Extended Shading Maps. In *Proc. of SmartGraphics*, pages 166–177, 2008.

[197] Steve M. F. Treavett and Min Chen. Pen-and-Ink Rendering in Volume Visualisation. In T. Ertl, B. Hamann, and A. Varshney, editors, *Proc. IEEE Visualization*, pages 203–210, 2000.

[198] V. D. Tsakanikas, L. K. Maichalis, D. I. Fotiadis, K. K. Naka, and C. V. Bourantas. *Intravascular Imaging: Current Applications and Research Developments*. IGI Global, 2012.

[199] Matthew van der Zwan, Wouter Lueks, Henk Bekker, and Tobias Isenberg. Illustrative molecular visualization with continuous abstraction. *Computer Graphics Forum*, 30(3):683–690, 2011.

[200] P. M. A. van Ooijen, R. J. M. van Geuns, B. J. W. M. Rensing, A. H. H. Bongaerts, P. J. de Feyter, and M. Oudkerk. Noninvasive Coronary Imaging Using Electron Beam CT: Surface Rendering Versus Volume Rendering. *American Journal of Roentgenology*, 180(1):223–226, 2003.

[201] Anna Vilanova, Bernhard Preim, Roy van Pelt, Rocco Gasteiger, Mathias Neugebauer, and Thomas Wischgoll. Visual Exploration of Simulated and Measured Blood Flow. *CoRR*, abs/1209.0999, 2012.

[202] Ivan Viola. *Importance-Driven Expressive Visualization*. PhD thesis, Institute of Computer Graphics and Algorithms, Vienna University of Technology, June 2005.

[203] Veronika Šoltészová, Daniel Patel, and Ivan Viola. Chromatic shadows for improved perception. In *Proc. of ACM SIGGRAPH*, pages 105–116, 2011.

[204] Leonard R Wanger, James Ferwerda, and Donald P Greenberg. Perceiving Spatial Relationships in Computer-Generated Images. *IEEE Tranactions on Computer Graphics and Applications*, 12(3):44–58, 1992.

[205] Max Wardetzky, Saurabh Mathur, Felix Kälberer, and Eitan Grinspun. Discrete laplace operators: no free lunch. In *SGP*, pages 33–37. Eurographics Association, 2007.

[206] Joseph R Weber. Proteinshader: illustrative rendering of macromolecules. *BMC Structural Biology*, 9:19, 5 2009.

[207] Georges Winkenbach and David H. Salesin. Computer-generated pen-and-ink illustration. In *Proc. of Computer Graphics and Interactive Techniques*, SIGGRAPH, pages 91–100, 1994.

[208] Georges Winkenbach and David H. Salesin. Rendering parametric surfaces in pen and ink. In *Proc. of Computer Graphics and Interactive Techniques*, SIGGRAPH, pages 469–476, 1996.

[209] Xuexiang Xie, Ying He, Feng Tian, Hock-Soon Seah, Xianfeng Gu, and Hong Qin. An effective illustrative visualization framework based on photic extremum lines (pels). *IEEE Transactions on Visualization and Computer Graphics*, 13:1328–1335, 2007.

[210] Stefan Zachow, Evgeny Gladilin, Robert Sader, and Hans-Florian Zeilhofer. Draw and cut: intuitive 3d osteotomy planning on polygonal bone models. In *CARS*, pages 362–369, 2003.

[211] Johannes Zander, Tobias Isenberg, Stefan Schlechtweg, and Thomas Strothotte. High quality hatching. *Computer Graphics Forum*, 23(3):421–430, 2004.

[212] Long Zhang, Ying He, and Hock Soon Seah. Real-time computation of photic extremum lines (PELs). *The Visual Computer*, 26 (6-8):399–407, 2010.

[213] Long Zhang, Ying He, Jiazhi Xia, Xuexiang Xie, and Wei Chen. Real-time shape illustration using laplacian lines. *IEEE Transactions on Visualization and Computer Graphics*, 17:993–1006, 2011.

[214] Kun Zhou, Qiming Hou, Rui Wang, and Baining Guo. Real-Time KD-Tree Construction on Graphics Hardware. In *Proc. of ACM SIGGRAPH Asia*, pages 126:1–126:11. ACM, 2008.

[215] S. Zhukov, A. Iones, and G. Kronin. An Ambient Light Illumination Model. In *Proc. of Eurographics Rendering Workshop*, pages 45–56, 1998.

[216] Sylvia Glaßer, Kai Lawonn, and Bernhard Preim. Visualization of 3D Cluster Results for Medical Tomographic Image Data. In *In Proc. of Conference on Computer Graphics Theory and Applications (VISIGRAPP/GRAPP)*, pages 169–176, 2014.

[217] Sylvia Glaßer, Kai Lawonn, Thomas Hoffmann, Martin Skalej, and Bernhard Preim. Combined Visualization of Wall Thickness and Wall Shear Stress for the Evaluation of Aneurysms. In *In Proc. of IEEE Transactions on Visualization and Computer Graphics*, (to appear), 2014.

[218] Paul Klemm, Kai Lawonn, Marko Rak, Bernhard Preim, Klaus Tönnies, Katrin Hegenscheid, Henry Völzke, and Steffen Oeltze. Visualization and Analysis of Lumbar Spine Canal Variability in Cohort Study Data. In *Proc. of Vision, Modeling and Visualization*, pages 121–128, 2013.

[219] Kai Lawonn, Rocco Gasteiger, and Bernhard Preim. Qualitative Evaluation of Feature Lines on Anatomical Surfaces. In *Bildverarbeitung für die Medizin (BVM)*, pages 187–192, 2013.

[220] Kai Lawonn, Rocco Gasteiger, and Bernhard Preim. Adaptive Surface Visualization of Vessels with Embedded Blood Flow Based on the Suggestive Contour Measure. In *Proc. of Vision, Modeling and Visualization*, pages 113–120, 2013.

[221] Kai Lawonn, Tobias Mönch, and Bernhard Preim. Streamlines for Illustrative Real-time Rendering. *Computer Graphics Forum*, 33(3):321–330, 2013.

[222] Kai Lawonn, Rocco Gasteiger, and Bernhard Preim. Adaptive Surface Visualization of Vessels with Animated Blood Flow. *Computer Graphics Forum*, page (minor revision), 2014.

[223] Kai Lawonn, Rocco Gasteiger, Christian Rössl, and Bernhard Preim. Adaptive and Robust Curve Smoothing on Surface Meshes. *Computer & Graphics*, 40(0):22 – 35, 2014.

[224] Kai Lawonn, Michael Krone, Thomas Ertl, and Bernhard Preim. Line Integral Convolution for Real-Time Illustration of Molecular Surface Shape and Salient Regions. *Computer Graphics Forum*, pages 181 – 190, 2014.

[225] Kai Lawonn, Patrick Saalfeld, and Bernhard Preim. Illustrative Visualization of Endoscopic Views. In *Bildverarbeitung für die Medizin (BVM)*, pages 276–281, 2014.

[226] Kai Lawonn and Bernhard Preim. Feature Lines for Illustrating Medical Surface Models: Mathematical Background and Survey. In *Visualization in Medicine and Life Sciences (VMLS)*, (in print), 2014.

[227] Tobias Mönch, Christoph Kubisch, Kai Lawonn, Rüdiger Westermann, and Bernhard Preim. Visually Guided Mesh Smoothing for Medical Applications. In *Proc. of VCBM - Eurographics Workshop on Visual Computing for Biology and Medicine*, pages 91–98, September 2012.

[228] Tobias Mönch, Kai Lawonn, Christoph Kubisch, Rüdiger Westermann, and Bernhard Preim. Interactive Mesh Smoothing for Medical Applications. *Computer Graphics Forum*, pages 1–12, 2013.

[229] Mathias Neugebauer, Kai Lawonn, Oliver Beuing, Philipp Berg, Gabor Janiga, and Bernhard Preim. AmniVis - A System for Qualitative Exploration of Near-Wall Hemodynamics in Cerebral Aneurysms. *Computer Graphics Forum*, 32(3):251–260, 2013.

[230] Mathias Neugebauer, Kai Lawonn, Oliver Beuing, and Bernhard Preim. Automatic Generation of Anatomic Characteristics from Cerebral Aneurysm Surface Models. *International Journal of Computer Assisted Radiology and Surgery*, 8(2):279–289, March 2013.

[231] Roy Van Pelt, Rocco Gasteiger, Kai Lawonn, Monique Meuschke, and Bernhard Preim. Comparative Blood Flow Visualization for Cerebral Aneurysm Treatment Assessment. *Computer Graphics Forum*, pages 131–140, 2014.

EHRENERKLÄRUNG

Ïch versichere hiermit, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; verwendete fremde und eigene Quellen sind als solche kenntlich gemacht. Insbesondere habe ich nicht die Hilfe eines kommerziellen Promotionsberaters in Anspruch genommen. Dritte haben von mir weder unmittelbar noch mittelbar geldwerte Leistungen für Arbeiten erhalten, die im Zusammenhang mit dem Inhalt der vorgelegten Dissertation stehen.

Ich habe insbesondere nicht wissentlich:

- Ergebnisse erfunden oder widersprüchliche Ergebnisse verschwiegen,
- statistische Verfahren absichtlich missbraucht, um Daten in ungerechtfertigter Weise zu interpretieren,
- fremde Ergebnisse oder Veröffentlichungen plagiiert,
- fremde Forschungsergebnisse verzerrt wiedergegeben.

Mir ist bekannt, dass Verstöße gegen das Urheberrecht Unterlassungs- und Schadensersatzansprüche des Urhebers sowie eine strafrechtliche Ahndung durch die Strafverfolgungsbehörden begründen kann. Die Arbeit wurde bisher weder im Inland noch im Ausland in gleicher oder ähnlicher Form als Dissertation eingereicht und ist als Ganzes auch noch nicht veröffentlicht."

Magdeburg, 15. September 2014

Kai René Hartmut Lawonn

CURRICULUM VITAE

Personal Details

Name:	**Kai Lawonn**
Email:	lawonn@isg.cs.uni-magdeburg.de
Date of birth:	04 October 1985
Place of birth:	Berlin
Civil status:	Single, no children
Nationality:	German

Education

04/2006 – 06/2011	**Diploma** Freie Universität Berlin (FU-Berlin) Course: Mathematics (minor subject Physics) Intermediate diploma: 1.0 Diploma: 1.2
08/1998 – 07/2005	**Abitur** Carl-Friedrich-Gauß-Gymnasium Advanced course: Mathematics and Physics Grade: 2.0
08/1992 – 07/1998	Carl-Humann-Grundschule

Work Experience

01/2014 -	Otto-von-Guericke-University: Research assistant in the visualization group of Prof. Preim
01/2012 – 12/2013	Dissertation fellowship from Sachsen-Anhalt in the visualization group of Prof. Preim
10/2011 – 12/2011	Otto-von-Guericke-University: Student assistant in the visualization group of Prof. Preim
02/2011 – 12/2012	Visage Imaging (Amira): Student assistant

06/2007 – 01/2011	FU - Berlin: Student assistant in the geometry processing group of Prof. Polthier
03/2007 – 12/2007	Studienkreis: Teaching of mathematics and physics
07/2006 – 06/2007	Schülerhilfe Wandlitz: Teaching of mathematics and physics
07/2007 – 12/2012	Lerntreff: Voluntary teaching of mathematics and physics for slow learner
01/2006 – 06/2007	Lerntreff: Teaching of mathematics and physics

Awards

2013	3. Place Best Paper Award (VMV)
2012	3. Place Best Paper Award (VCBM)

Magdeburg, 15. September 2014