Parametric Methods for Image Processing Using Active Contours With Topology Changes

Dissertation

zur Erlangung des Doktorgrades
der Naturwissenschaften (Dr. rer. nat.)
der Fakultät für Mathematik
der Universität Regensburg

vorgelegt von
Heike Benninghoff
aus Nabburg
im Jahr 2015

Promotionsgesuch eingereicht am 04. Februar 2015.

Die Arbeit wurde angeleitet von Prof. Dr. Harald Garcke.

Prüfungsausschuss:	Vorsitzender:	Prof. Dr. Klaus Künnemann
	1. Gutachter:	Prof. Dr. Harald Garcke
	2. Gutachter:	Dr. Robert Nürnberg, Imperial College London
	weiterer Prüfer:	Prof. Dr. Georg Dolzmann

Abstract

In this thesis we consider parametric methods for image processing based on active contours. We introduce an efficient scheme for image segmentation by evolving parametric hypersurfaces. More precisely, we present methods for segmentation of 1) two-dimensional, planar images, of 2) images on curved surfaces and of 3) three-dimensional images.

The developed methods can handle complex curve networks with possible triple junctions and intersections of the curves with the image boundary. Also curves with free endpoints are supported. The methods can be used to segment a given image in regions of arbitrary number, separated by hypersurfaces.

Numerically, the evolving curves and surfaces are discretized and the resulting schemes are solved by finite differences and finite elements. We show that the parametric approach for curve evolution in the plane and on surfaces has good properties concerning the equidistribution of mesh points along the discretized curves. For evolving surfaces, we observe problems with the quality of the triangulated meshes in rare cases only. We propose a method for an efficient mesh regularization which is incorporated into the evolution scheme for surfaces.

Standard parametric approaches cannot automatically handle topology changes like splitting and merging of curves and surfaces, creating and deleting triple junctions and boundary intersection points of curves as well as changing the genus of a surface. Therefore, we introduce an efficient method to detect and execute such topology changes. Using our approach, the computational effort to detect a topology change depends only linearly on the number of mesh points.

In addition to image segmentation, we propose a method for edge-preserving image smoothing. The denoising of the image is executed as a postprocessing step, subsequently to the segmentation. Thereby, diffusion equations with Neumann boundary conditions are solved in the already segmented regions. In the case of images defined on surfaces, this results in partial differential equations on manifolds.

Finally, we demonstrate the developed methods on various artificial and real images and show the efficiency of the methods and their application to real, practical image processing tasks arising in medicine, navigation, Earth observation and in many other areas.

Zusammenfassung

In dieser Arbeit betrachten wir parametrische Methoden zur Bildverarbeitung mit Aktiven Konturen. Wir führen ein effizientes Schema zur Bildsegmentierung durch evolvierende parametrische Hyperflächen ein. Im Detail stellen wir Methoden vor zur Segmentierung von 1) zweidimensionalen, ebenen Bildern, von 2) Bildern auf gekrümmten Flächen und von 3) dreidimensionalen Bildern.

Die entwickelten Methoden können komplexe Kurvennetzwerke mit möglichen Tripelpunkten und Schnitten der Kurven mit dem Bildrand handhaben. Auch Kurven mit freien Randpunkten werden unterstützt. Die Verfahren können verwendet werden, um ein gegebenes Bild in beliebig viele Regionen zu segmentieren, die durch Hyperflächen voneinander getrennt sind.

Numerisch betrachtet, werden die evolvierenden Kurven und Flächen diskretisiert und die dabei entstehenden Schemata werden durch Finite Differenzen und Finite Element Methoden gelöst. Wir zeigen, dass der parametrische Ansatz für Kurvenevolution in der Ebene und auf Flächen gute Eigenschaften bezüglich der Gleichverteilung von Gitterpunkten entlang der diskretisierten Kurven hat. Bei evolvierenden Flächen stellen wir Probleme mit der Qualität triangulierter Gitter nur in seltenen Fällen fest. Wir schlagen eine Methode zur effizienten Gitterregularisierung vor, die in das Schema zur Flächenevolution eingearbeitet ist.

Übliche parametrische Ansätze können Topologieänderungen wie Aufspalten und Verschmelzen von Kurven, Erstellen und Entfernen von Tripelpunkten und Schnittpunkten von Kurven mit dem Bildrand, sowie Ändern des Geschlechts einer Fläche, nicht automatisch handhaben. Daher führen wir ein effizientes Verfahren ein, um solche Topologieänderungen detektieren und ausführen zu können. Mit unserem Ansatz hängt der Rechenaufwand eine Topologieänderung zu detektieren nur linear von der Anzahl der Gitterpunkte ab.

Zusätzlich zu Bildsegmentierung schlagen wir eine Methode für kantenerhaltende Bildglättung vor. Das Entrauschen des Bildes wird als Nachprozessierungschritt nach der Segmentation ausgeführt. Dabei werden Diffusionsgleichungen mit Neumann-Randbedingungen in den bereits segmentierten Regionen gelöst. Im Fall von Bildern auf Flächen führt dies zu Partiellen Differentialgleichungen auf Mannigfaltigkeiten.

Schließlich demonstrieren wir die entwickelten Methoden anhand von verschiedenen künstlichen und realen Bildern und zeigen die Effizienz der Verfahren als auch ihre Anwendbarkeit auf reale, praktische Bildverarbeitungsaufgaben, die in der Medizin, der Navigation, der Erdbeobachtung und in vielen anderen Gebieten auftreten.

Acknowledgments

My sincere thanks go to my advisor Prof. Dr. Harald Garcke for his valuable advice and guidance throughout the last years. I am very grateful for his support of my PhD project, and for the motivating meetings and discussions. The joint publication Benninghoff and Garcke (2014) „*Efficient image segmentation and restoration using parametric curve evolution with junctions and topology changes*“, published in SIAM Journal on Imaging Sciences 7(3), was very helpful. Furthermore, I want to thank Dr. Robert Nürnberg, Imperial College London, for the second review of this thesis. I am grateful to Prof. Dr. Christian Stroszczynski, Director, Department of Radiology of University Hospital Regensburg, for providing a large data set of computed tomography images which has been extremely helpful in testing the algorithm for segmentation of 3D images. Last but not least, I'd like to thank my family for their general support and my partner Dominik for his constant support and for proof-reading of the thesis. Working on a PhD thesis in mathematics in parallel to a regular employment at German Aerospace Center (DLR) was sometimes quite challenging. Therefore I am grateful to all people who supported me during this time.

Contents

Chapter 1

Introduction

Image processing covers a variety of applications, tasks and methods and gained more and more importance in the last decades due to rapid developments of digital cameras, optical devices and sensors in consumer and professional fields. Also developments in the field of computing allow for fast image processing.

One major challenge in computer vision is the autonomous detection of objects in given image data without or with little user intervention. The number of objects is often not known in advance. This demands appropriate image processing techniques which can handle arbitrary scenes and compositions of objects in images. Automated image processing plays an important role in many engineering fields. Autonomous applications require novel methods and advanced algorithms compared to user-guided image processing.

Furthermore, the processing of large 3D image data poses another great challenge. Such demands arise in medical areas like radiology where for example 3D image data from computed tomography should be processed.

According to Aubert and Kornprobst (2006), three main approaches exist for image analysis and image processing. These are 1) stochastic modeling based on Markov theory, 2) Wavelets coming from classical signal processing and 3) image processing based on partial differential equations (PDEs).

In this thesis, we focus on PDE based image processing. Image processing problems can be often formulated as optimization problems, where a certain energy functional should be minimized. Calculus of variations, partial differential equations and numerical mathematics provide the necessary mathematical framework and tools to describe and solve the problems which arise in image processing. Here, we consider two tasks: image *restoration* and image *segmentation* which are two important sub-areas in image processing.

Images are typically affected by noise and artifacts caused by the acquisition system or by the transmission of the digital data. Both deterministic and random noise can degrade the image quality. Image restoration aims at removing or reducing the image noise. Recovering the original image from a given, noise-affected image results in an inverse, often ill-posed problem, see Aubert and Kornprobst (2006).

Let $u_0 : \Omega \to \mathbb{R}$, $\Omega \subset \mathbb{R}^2$, be a given, noise-affected, two-dimensional image and let $u_{\text{ideal}} : \Omega \to \mathbb{R}$ be the noise-free, ideal image. We call $\Omega \subset \mathbb{R}^2$ the *image domain*. Let us assume a simple degradation model

$$u_0 = u_{\text{ideal}} + \eta, \tag{1.1}$$

where η is some additive Gaussian noise with zero mean in Ω. This degradation model is a strong simplification and real image noise may require a more advanced noise model. However, the model (1.1) can be used to develop and understand some basics of image restoration.

An approximation u of the unknown image u_{ideal} can be obtained by minimizing

$$E(u) = \lambda \int_\Omega (u - u_0)^2 \, \mathrm{d}x + \int_\Omega \|\nabla u\|^2 \, \mathrm{d}x. \tag{1.2}$$

The energy contains two competing terms: The first integral will be small if u is close to u_0. The second integral will be small if u changes only slightly in the image domain. The second term is added for regularization purposes and provides smoothness of the solution u. The constant $\lambda > 0$ is a weighting parameter.

We search for a minimizer $u \in W^{1,2}(\Omega)$, i.e. u and its weak derivative ∇u are in $L^2(\Omega)$ and $[L^2(\Omega)]^2$, respectively. Using methods of the theory of calculus of variations, we obtain the corresponding Euler-Lagrange equation

$$-\frac{1}{\lambda}\Delta u + u = u_0, \quad \text{in } \Omega, \tag{1.3a}$$

with the Neumann boundary condition

$$\frac{\partial u}{\partial \vec{n}_{\partial\Omega}} = 0, \quad \text{on } \partial\Omega, \tag{1.3b}$$

where $\vec{n}_{\partial\Omega}$ is a normal vector field on the boundary $\partial\Omega$.

Another classical smoothing technique uses the heat equation. This smoothing method is based on the idea that an image can be embedded in a one-parameter family of images, see Koenderink (1984), with the resolution of the image as parameter. In detail, we consider a family $u(\,.\,,t) : \mathbb{R}^2 \to \mathbb{R}$, $t \geq 0$ with $u(\,.\,,0) = u_0$ being the original image $u_0 : \mathbb{R}^2 \to \mathbb{R}$. For $t > 0$, $u(\,.\,,t)$ is a smoothed version of u_0 obtained by convolution of the original image u_0 with a Gaussian smoothing kernel Φ_t given by $\Phi_t(\vec{x}) = \frac{1}{4\pi t} \exp(-1/(4t)\, \vec{x}^T \vec{x})$. Thus, u satisfies the heat equation

$$u_t - \Delta u = 0. \tag{1.4}$$

Both restoration techniques (1.3) and (1.4) are based on the smoothing effect of the Laplace operator. However, the Laplace operator has the undesired behavior that edges in the image are not preserved but smoothed out. A better image denoising can be obtained by considering *anisotropic diffusion* which enhances the edges, as proposed by Perona and Malik (1990): For that, Δu is replaced by $\operatorname{div}(f \nabla u)$, where $f : \Omega \to \mathbb{R}$ is an *edge indicator* function. Edges can be characterized by the image gradient, since $\|\nabla u\|$ is locally high at edges, or alternatively by the jump set of the given image u_0. Typically, an edge indicator function f is designed such that it is small where $\|\nabla u\|$ is large in order to suppress the diffusion at edges. For example, we could set $f(\vec{x}) = \exp(-a\|\nabla u(\vec{x})\|^2)$ with a constant $a > 0$ or $f(\vec{x}) = 1/\left(1 + a\|\nabla u(\vec{x})\|^2\right)$ for $\vec{x} \in \Omega$, see Perona and Malik (1990).

Figure 1.1 shows an exemplary, noisy image and two denoising approaches. The first approach (Figure 1.1, center) uses (1.3) for smoothing the image. As expected, the edges in the image are strongly smoothed out. The second approach (Figure 1.1, right) performs an edge-preserving image smoothing which will be developed in this thesis.

Other alternative smoothing methods are based on the principle of *total variation*, see Rudin et al. (1992), where the L^1-norm of the gradient of u is used for the energy to be minimized instead of the L^2-norm, cf. (1.2). This idea is also used for example by Chan

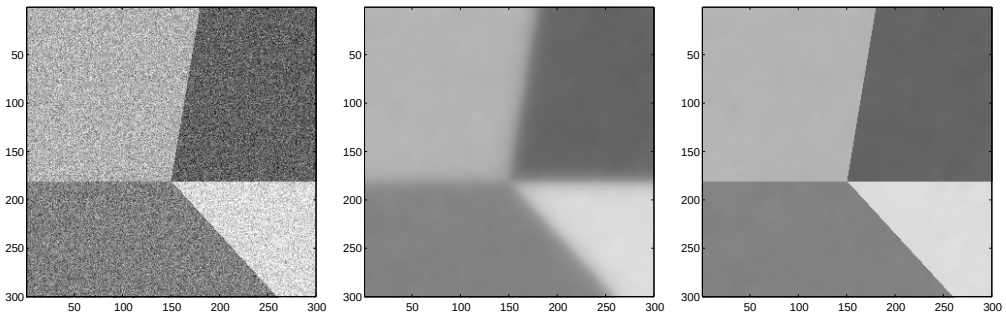


Figure 1.1: Example of a noisy image (left), a denoised version based on (1.3) with blurred edges (center) and a denoised version with preserved edges (right).

et al. (2001). More generally, the L^p-norm for $1 \leq p \leq 2$ is considered by Tang et al. (2002). Robust anisotropic diffusion has been investigated by Black et al. (1998), where the authors consider several choices of a robust, anisotropic error norm. In the context of image denoising/smoothing and image enhancement we also mention Yu and Bajaj (2002) for diffusion schemes for vector-valued images like color images and refer to Scherzer et al. (2009) for noise models and denoising techniques.

Apart from degradation based on noise, images can also be affected by so-called *inpainting* which can be removed by special techniques, see Bertalmio et al. (2000); Chan et al. (2002) among others. Inpainting is not covered in this thesis.

Image restoration is an important processing step; often it is performed before more advanced and application-dependent techniques are applied on the images. Beside image restoration, image segmentation is another classical image processing area.

Image segmentation aims at dividing an image into characteristic subsets, so-called *regions* or *phases*. The phases typically represent foreground objects or the background of an image. The regions can be identified by their boundaries, where the gray or color value rapidly changes (see e.g. Kass et al. (1988); Malladi et al. (1995); Caselles et al. (1997a)), or they can be characterized by their mean gray value or mean color, by their texture or by some other grouping (see e.g. Mumford and Shah (1989); Ronfard (1994); Chan and Vese (2001); Tsai et al. (2001)). The different segmentation methods can thus be divided into two main groups: *edge-based* image segmentation and *region-based* image segmentation.

A very popular approach for image segmentation and edge-detection is the *active contours* method, originally developed by Kass et al. (1988). One or more curves, also called *contours*, evolve in the image domain and stop locally at region boundaries or edges. The motion of the curves can be described by *evolution equations*, which aim to minimize certain energies. The involved energies typically contain an internal energy, which controls the smoothness of the curve, and an external energy, which pushes the contours to edges or region boundaries. The internal energy is often based on length or area terms. The length of the contours or the enclosed area will only grow if the external energy is large enough. Thus, the segmentation method can be adapted to be robust with respect to moderate noise. The external energy is typically designed such that it is small where the gradient of the image function is high (edge-based methods) or such that the image function or some related quantity changes only slightly inside the enclosed regions (region-based methods).

In this thesis, we mainly consider flow of curves based on the functional of Mumford and Shah (1989): Let $u_0 : \Omega \to \mathbb{R}$ be a given image function. We search for a set of curves

$\Gamma = \Gamma_1 \cup \dots \Gamma_{N_C}$, $N_C \in \mathbb{N}$, and a piecewise smooth function $u : \Omega \to \mathbb{R}$ with possible discontinuities across Γ approximating the original image u_0. The energy to be minimized is

$$E(u, \Gamma) = \sigma|\Gamma| + \int_{\Omega \setminus \Gamma} \|\nabla u\|^2 \, \mathrm{d}x + \lambda \int_{\Omega} (u_0 - u)^2 \, \mathrm{d}x, \tag{1.5}$$

where $\sigma, \lambda > 0$ are weighting parameters and $|\Gamma|$ denotes the total length of the curve. As discussed above, the length term provides smoothness of the curves Γ, the second term ensures that u has no large gradients in $\Omega \setminus \Gamma$ and the third term ensures that the approximation u is close to the given image u_0.

The Mumford-Shah functional can be used for both image segmentation and image restoration since its minimizer (u, Γ) provides a denoised image u and a segmentation of the image given by the curves belonging to Γ. An important variant of the original problem is the restriction to piecewise constant approximations u, called *minimal partition problem*, see Chan and Vese (2001).

Image segmentation is a classical area in image analysis, see Kass et al. (1988); Mumford and Shah (1989); Ronfard (1994), but still significant in more present research, see e.g. Chan and Vese (2001); Tsai et al. (2001); Beneš et al. (2004); Doğan et al. (2008); Chung and Vese (2009); Mille (2009); Arbelaez et al. (2011); Chambolle et al. (2012) to mention a few selected works. There is also a variety of related image processing tasks like *object detection* (Paragios and Deriche, 2000; Fergus et al., 2003) or *pattern recognition* (Bezdek et al., 2005)), *feature extraction* (Nixon and Aguado, 2002), *anomaly detection* (Chandola et al., 2009) and *tracking* of objects in sequences of images (Paragios and Deriche, 2000; Moelich and Chan, 2003).

Image processing has many applications in different areas like for example engineering and medicine. One large sector in engineering is optical navigation, for example autonomous, visual navigation in automotive engineering (Gavrila and Philomin, 1999) or in avionics (Dobrokhodov et al., 2006), navigation and path planning in robotics and machine vision (DeSouza and Kak, 2002; Bonin-Font et al., 2008), as well as autonomous navigation in space (Bhaskaran et al., 1996; Woffinden and Geller, 2007; Benninghoff et al., 2014). Image processing plays an important role in Earth observation and remote sensing (Jensen, 1996; Tuia et al., 2012; Čunderlík et al., 2013). As mentioned above, in radiology, image processing is applied for analysis of images which are generated by e.g. computed tomography or magnetic resonance imaging (Udupa and Herman, 1999; Ardon et al., 2005; Li et al., 2005; Rousseau and Bourgault, 2009; Sharma and Aggarwal, 2010). Image processing and image analysis are also applied in industry for quality control in manufacturing and for automated visual inspection (Chin and Harlow, 1982; Du and Sun, 2004).

The image segmentation methods, that we consider, use evolution of hypersurfaces. The resulting evolution equations, derived from the Mumford-Shah functional or minimal partition problems, can be written as parabolic PDEs for a parametrization of the hypersurfaces. The segmentation result is used as input for the restoration technique to obtain an edge-preserving smoothing (cf. Figure 1.1, right). The corresponding diffusion equations are elliptic PDEs for the image approximation.

We will consider three different classes of images: First, we consider classical 2D images generated by an image acquisition system like a digital camera, see Gonzalez and Woods (2001). 2D images can be mathematically described by an image function $u_0 : \Omega \to \mathbb{R}^{(d)}$, where $\Omega \subset \mathbb{R}^2$ is a two-dimensional domain. For gray-scaled images, we have $d = 1$, for color images, we have $d \geq 1$. For example, if the image is given in RGB (red, green, blue) color intensities, we have $d = 3$. Second, we consider images on non-flat surfaces. These images

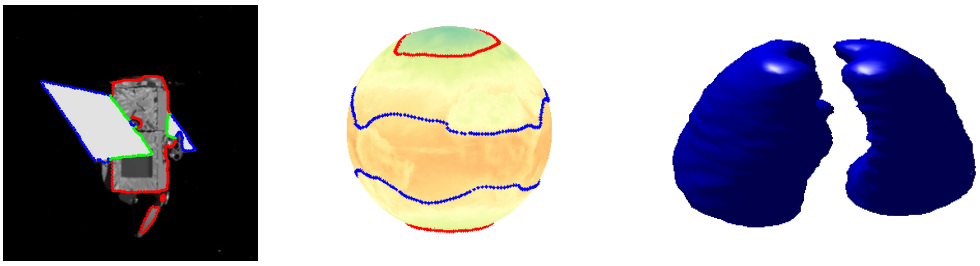


Figure 1.2: Exemplary images and segmentations of three different classes of images. Left: 2D image from visual navigation showing an image of a satellite and the detected region boundaries. Center: Image defined on the Earth's surface visualizing the distribution of the Earth's net radiation and the detected region boundaries. Right: Extracted region boundaries of a 3D medical image (lung segmentation). Image courtesy: Left: Astos Solutions GmbH (2013), Center: NASA Earth Observation data set, NASA (2014), Right: C. Stroszczynski, Radiology, University Hospital Regensburg.

can be generated by some reconstruction, for example by using 3D scans, see e.g. Paysan et al. (2009), and/or by composing 2D images of a 3D object from different viewing angles to a non-flat image. For example in the case of Earth observation images, global Earth data can be obtained by composing several 2D images of single parts of the Earth's surface. An image defined on a surface can be described by an image function $u_0 : \mathcal{M} \to \mathbb{R}^{(d)}$, where $\mathcal{M} \subset \mathbb{R}^3$ is a two-dimensional manifold. Third, we consider 3D images given by $u_0 : \Omega \to \mathbb{R}^{(d)}$, where $\Omega \subset \mathbb{R}^3$ is a three-dimensional domain. In medicine, a 3D image is often generated by a set of 2D tomographic slice images. Figure 1.2 shows examples of a segmentation of a 2D image (left), a segmentation of an image defined on a surface (center) and the detected object boundaries of a 3D image (right).

During the evolution of curves or surfaces, topology changes like splitting or merging can occur, since the number and the location of the objects in the image is often not known in advance. Because of its ability to handle topology changes automatically, the level set technique of Osher and Sethian (1988) is used by many authors to solve the evolution equations which arise from the active contours method, see e.g. Caselles et al. (1997a), Kichenassamy et al. (1996), Chan and Vese (2001), Tsai et al. (2001), Sapiro (2006). There, a hypersurface is embedded as the zero level set of a function defined on the image domain Ω.

In this thesis, we propose an alternative, parametric method for image segmentation which is based on the works of Barrett et al. (2007a, 2008a,b, 2010a). Using a parametric scheme, the image segmentation problem results in a one-dimensional problem in the case of 2D images and in the case of images on 2D surfaces, since we consider evolving curves. The segmentation problem results in a two-dimensional problem in the case of 3D images, since we consider evolving surfaces. Using standard level set methods, one has to solve two-dimensional problems in the case of 2D images and three-dimensional problems in the case of 3D images, since the level set function is defined on the 2D or 3D image domain. Curves on surfaces cannot be directly represented by the zero level set of only one level set function defined in a 3D neighborhood of the surface. They can be handled by intersection of the zero level sets of two such level set functions, see Cheng et al. (2002); Krüger et al. (2008). Additionally, we can also handle triple junctions and curves with free endpoints (also called crack-tips or open boundaries) with the parametric approach.

The method of Barrett et al. (2007a, 2010a) further provides a good mesh quality of the discretized curves and surfaces. In general, using a parametric scheme, node points on a curve can locally bunch together and can have large distances at other locations along the curve. For planar curves and for curves on surfaces, we can prove equidistribution of mesh points along the curve for a semidiscrete scheme using the method of Barrett et al. (2007a, 2010a). Also in the fully discrete case, we obtain nearly an equidistribution of curve mesh points. Evolving surfaces are discretized by triangulated meshes. We observe problems with the mesh quality in extreme cases only, for example immediately before a splitting of a surface into two surfaces occurs. By a slight modification of the parametric scheme for surfaces, we can control the tangential motion of the vertices of the triangulation and can thus improve the mesh quality without the need of an additional algorithm for mesh regularization.

Using a parametric scheme, topology changes cannot be handled automatically which is often considered as the main drawback of parametric methods. In this thesis, we extend the method of Mikula and Urbán (2012) for detection of topology changes of curves, such that also topology changes involving triple junctions and intersections with the image boundary can be dealt with. The method of Mikula and Urbán (2012) is based on an auxiliary 2D background grid to detect topology changes efficiently. A topology change may occur if nodes from different parts of the curve or nodes from different curves are located in one grid element. Further, we extend the method developed for 2D images to higher dimensions, such that topology changes of evolving curves on surfaces and of evolving surfaces in $\mathbb{R}^3$ can be detected and executed. For that, we introduce an auxiliary 3D background grid. Apart from splitting or merging of surfaces, a new class of topology changes can occur in the 3D case: The genus of a surface can increase or decrease, for example, when a sphere evolves to a torus or vice versa. The detection of topology changes can be performed efficiently using the idea of Mikula and Urbán (2012): If N is the number of nodes or vertices of the discretized curves or surfaces, the search for topology changes has an effort of $\mathcal{O}(N)$.

Figure 1.3 shows two merging curves and visualizes the parametric and the level set method. The topology change need not be detected explicitly using the level set method. However, a higher-dimensional problem has to be solved at each time step of the evolution. The effort of standard level set methods is dependent on the size and discretization of the two-dimensional image domain, whereas the parametric scheme mainly depends on the one-dimensional curves and their discretizations, which typically results in a much smaller effort. Furthermore, to obtain the final 1D contour (see right subfigures in Figure 1.3), the zero level set has to be first reconstructed from the 2D level set function.

Let us summarize the main contributions of this thesis:

- A *parametric* method based on Barrett et al. (2007a), originally developed for mean curvature flow of curves and related flows, is extended to image segmentation problems. The segmentation technique is based on the method of Mumford and Shah (1989) and Chan and Vese (2001), and can therefore segment images with sharp and weak edges. The method results in an efficient segmentation including *multiple phases*, possible *junctions* and *intersections with the image boundary*.

- The segmentation technique is extended such that also non-interface curves, i.e. curves with so-called *free endpoints*, can be dealt with.

- The method of Mikula and Urbán (2012) is extended to detect and execute *topology changes* such as splitting, merging, creation of triple junctions, creation of boundary intersection points.

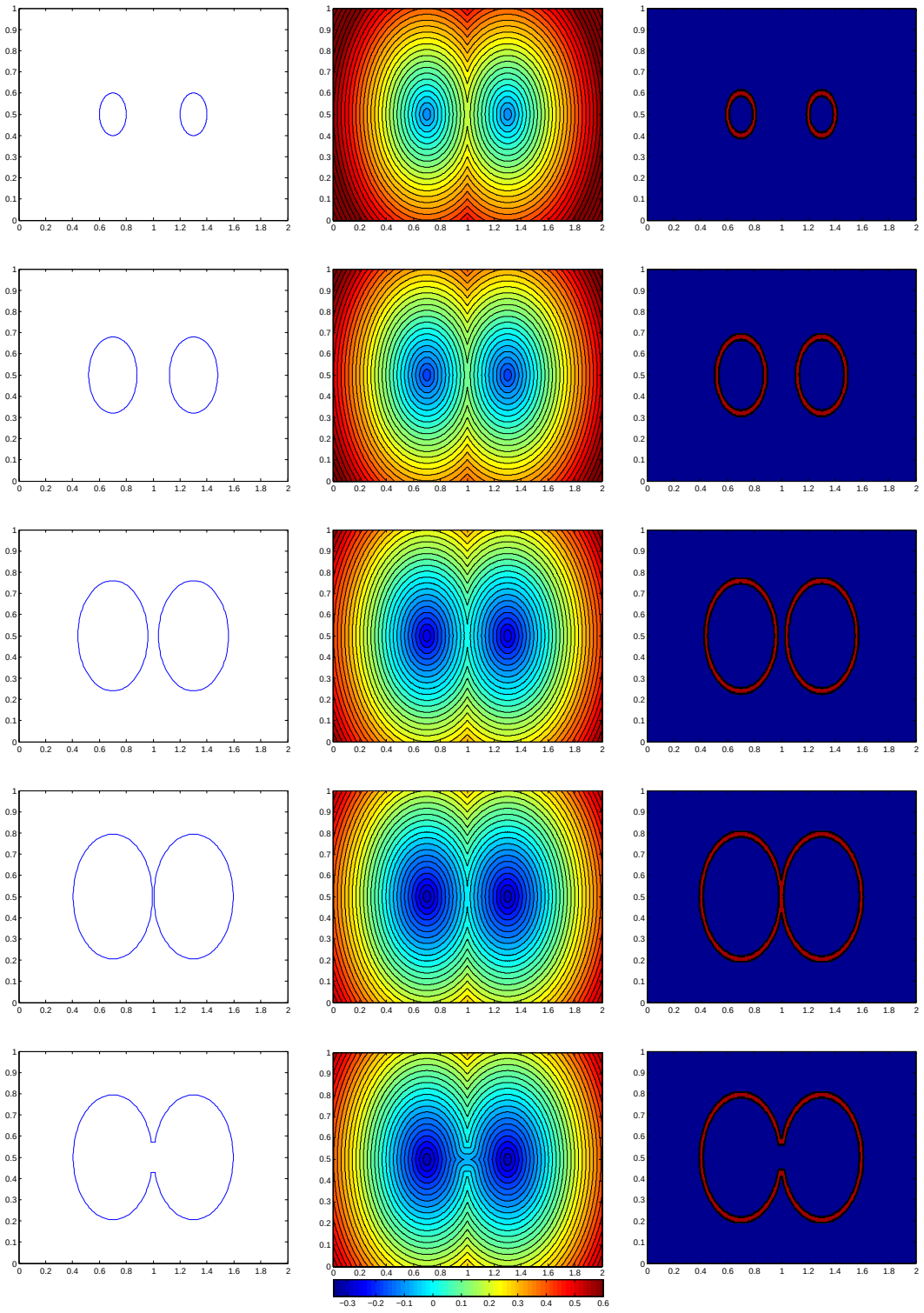


Figure 1.3: Merging of two curves. Left: Parametric approach. The topology change must be detected and handled by a special sub-routine. Center: Level set function. Right: Extracted zero level set. In the level set method, the curve set is embedded as zero level set of a two-dimensional function. The topology change is handled automatically. However, for the level set approach, an equation on a two-dimensional domain has to be solved at each time step of the evolution, whereas for the parametric approach, only a one-dimensional problem has to be solved.

- Many control features of the segmentation algorithm can be automated. Examples are *adaptive control* and *automated setting* of e.g. the time step size, the weight of the internal energy, and the grid size of the auxiliary background grid used to detect topology changes.

- Image denoising is computed as postprocessing step using the identified regions, which have already been detected during the image segmentation process. With the knowledge about regions, an *edge-preserving denoising* can be performed.

- Segmentation and denoising of *vector-valued images*, in particular of color images, is supported. For some color spaces, we solve constrained minimization problems.

- Images defined on *non-flat surfaces* are segmented and denoised. As in Barrett et al. (2010a), an additional condition needs to be concerned, such that the curves in $\mathbb{R}^3$ remain on the surfaces during their evolution. The ability to handle complex curve networks including junctions and topology changes as well as vector-valued image data is extended from the planar case to the non-planar case. For image restoration, we make use of the Laplace-Beltrami operator to smooth the image data.

- *3D images* are segmented by extending the image segmentation method from evolving curves to evolving *surfaces*. The method includes the possibility to handle complex topology changes of surfaces such as splitting, merging, increase and decrease of the genus of the surface.

- The developed algorithms are applied on artificial and real image data. The *real image data* is chosen from different *applications* like photo analysis, tracking/navigation, medicine and Earth observation. For 3D image segmentation, a dataset of computed tomography images provided by the radiology department of the University Hospital Regensburg is processed, where we also discuss practical problems like non-Gaussian noise in medical images.

Outline of the thesis

The remaining part of the thesis is structured as follows:

In Chapter 2, we collect some mathematical basics, definitions and theorems concerning curves and surfaces.

In Chapter 3, we consider image segmentation and denoising of two-dimensional images. We start with an overview and introduction to active contours, and consider several energy minimization problems from edge-based and region-based segmentation. Using the theory of calculus of variations, we derive the corresponding evolution equations for the curves from the energy functionals. The analytical aspects of the parametric method including multiple phases, junctions and free endpoints are described in detail. We also derive the image denoising scheme and mention how colored images can be handled. This is followed by a presentation of the numerical scheme, which can be derived both from a finite element and a finite difference approach. We further consider a semidiscrete scheme for which we can prove equidistribution of the mesh points. We describe our method to handle topology changes in 2D, state a finite difference approach to solve the image denoising scheme with Neumann boundary conditions and present the necessary adaptations for curves with free endpoints. We finish the chapter by presenting various experiments including a demonstration of object tracking in a sequence of images.

In Chapter 4, we consider segmentation and denoising of images defined on non-flat surfaces. We consider edge-based and region-based active contours on surfaces. For curves on surfaces the method of the two-dimensional case is slightly modified. Special function spaces are considered which allow the curves to move only in tangential direction on the surfaces. We derive an image denoising scheme, where the Laplace operator of the 2D case is replaced by the Laplace-Beltrami operator for image diffusion on surfaces. We present the numerical schemes based on finite elements and consider a semidiscrete scheme for which we can prove equidistribution of the mesh points of the discretized curves. We describe how topology changes concerning curves on surfaces are handled. For that we make use of a 3D background grid together with a tubular neighborhood theorem. We present results from some experiments using artificial and real image data.

In Chapter 5, we present how the method can be extended to segmentation of 3D images. We shortly describe the analytical framework for region-based segmentation using evolving surfaces. The main focus in this chapter is the numerical scheme based on an approximation of the smooth surfaces by triangulated surfaces. The finite element approach and the discrete linear system are presented. We describe in detail our algorithm to detect and perform topology changes. The detection of topology changes is a simple extension of the 2D case, using an auxiliary 3D background grid instead of a 2D grid. The algorithm to finally execute the topology changes in 3D differs strongly from the 2D case. Vertices, edges and simplices have to be adapted when a topology change of a triangulated surface occurs. Often simplices have to be deleted at locations where a topology change occurs, resulting in intermediate holes. New simplices need to be created to close open holes of the surfaces. We further use a mesh regularization technique to control the tangential motion of the vertices on the surface, see Barrett et al. (2008c). We finish the chapter on 3D image segmentation by demonstrating various topology changes of surfaces using artificial images, followed by segmentation of real, medical image data from computed tomography.

A final conclusion and summary as well as a short discussion of additional research beyond the scope of this thesis are given in Chapter 6.

Chapter 2

Mathematical Basics

In this thesis, evolving curves in $\mathbb{R}^2$, evolving curves on two-dimensional manifolds and evolving surfaces in $\mathbb{R}^3$ are considered for image segmentation purposes. In this chapter, we give an introduction to hypersurfaces in $\mathbb{R}^d$ and curves on surfaces. We will state the main important definitions and theorems for hypersurfaces. This chapter covers tangential and normal vector fields, integration, first and second fundamental form, tangential derivatives and surface divergence, Laplace-Beltrami operator and the definition of mean curvature via the Weingarten map. We shortly introduce several methods to represent hypersurfaces: these are direct methods like parametric or graph-based approaches and indirect methods like level set or phase field approaches. Furthermore, we consider evolving hypersurfaces and state a transport theorem, a basic theorem to which we often refer in this thesis. Finally, we consider curves on surfaces and introduce important concepts and definitions like geodesic curvature and flows of curves on surfaces.

2.1 Hypersurfaces

We consider smooth hypersurfaces $\Gamma \subset \mathbb{R}^d$, i.e. smooth submanifolds of $\mathbb{R}^d$ of dimension $d-1$. For the theory of smooth manifolds including definitions of (sub)manifolds, tangent spaces and integration on manifolds, we refer to Lee (2002), Lang (2002) and Jänich (2005).

2.1.1 Basic Definitions and Theorems

In this section, we present the main basic definitions and theorems about hypersurfaces which are used in the following chapters of this thesis. The main references are Kühnel (2005), Eschenburg and Jost (2007), Eck et al. (2008), Deckelnick et al. (2005) and Dziuk and Elliott (2013).

Definition 2.1 (Regular Hypersurface)**.** *Let $U \subset \mathbb{R}^{d-1}$ be an open subset. A* regular parameterized hypersurface *is an immersion*

$$\vec{x} : U \to \mathbb{R}^d, \quad (u_1, \dots, u_{d-1}) \mapsto \vec{x}(u_1, \dots, u_{d-1}). \tag{2.1}$$

An unparameterized hypersurface *is an equivalence class of parameterized hypersurfaces, where $\vec{x} : U \to \mathbb{R}^d$ and $\vec{y} : V \to \mathbb{R}^d$ are equivalent if a diffeomorphism $\vec{\phi} : V \to U$ exists satisfying $\vec{y} = \vec{x} \circ \vec{\phi}$.*

In this thesis, we consider in particular the cases $d = 2$ and $d = 3$, i.e. curves in $\mathbb{R}^2$ and surfaces in $\mathbb{R}^3$.

Remark 2.2. *(i) Let $\vec{x} : U \to \mathbb{R}^d$ be a regular parameterized hypersurface and $\Gamma := \vec{x}(U)$. In the following, we also call Γ a* regular hypersurface. *We call Γ a* smooth hypersurface *if $\vec{x}$ is of class C^∞.*

(ii) We extend the term and call $\Gamma \subset \mathbb{R}^d$ a smooth hypersurface if local *smooth parameterizations exist. I.e. we require that for each point $\vec{p} \in \Gamma$, there exist a neighborhood $U_\Gamma \subset \Gamma$ of $\vec{p}$ and an immersion $\vec{x} : U \to \mathbb{R}^d$ with $\vec{x}(U) = U_\Gamma$. Thus, facts for hypersurfaces with global smooth parameterizations can be generalized by using a partition of unity and local smooth parameterizations.*

Now, we consider functions which are defined on hypersurfaces and define the term *differentiability*:

Definition 2.3 (Differentiable Functions)**.** *Let $\Gamma \subset \mathbb{R}^d$ be a smooth hypersurface with a smooth parameterization $\vec{x} : U \to \mathbb{R}^d$. A function $f : \Gamma \to \mathbb{R}^m$ is called C^k-function, $k \in \mathbb{N}$, if $f \circ \vec{x} : U \to \mathbb{R}^m$ is of class C^k.*

For local considerations, local coordinate systems are necessary. The main local directions are given by so-called tangential and normal vector fields:

Definition 2.4 (Tangential and Normal Spaces)**.** *Let $\Gamma \subset \mathbb{R}^d$ be a smooth hypersurface with a parameterization $\vec{x} : U \to \mathbb{R}^d$. Let $\vec{u} \in U$ and $\vec{p} = \vec{x}(\vec{u}) \in \Gamma$. The vectors $\frac{\partial \vec{x}}{\partial u_i}(\vec{u})$, $i = 1, \dots, d-1$, span a $(d-1)$-dimensional space of so-called* tangent vectors

$$T_{\vec{p}}\Gamma = \operatorname{span}\left\{ \frac{\partial \vec{x}}{\partial u_1}(\vec{u}), \dots, \frac{\partial \vec{x}}{\partial u_{d-1}}(\vec{u}) \right\}. \tag{2.2}$$

The space $T_{\vec{p}}\Gamma$ is called tangential space of Γ at $\vec{p}$*. Its orthogonal complement $N_{\vec{p}}\Gamma = (T_{\vec{p}}\Gamma)^\perp$ in $\mathbb{R}^d$ is called* normal vector space of Γ at $\vec{p}$*.*

Definition 2.5 (Orientable Hypersurfaces)**.** *A hypersurface Γ is called* orientable*, if a continuous normal vector field $\vec{\nu} : \Gamma \to S^{d-1}$ exists, where S^{d-1} is the $(d-1)$-dimensional sphere in $\mathbb{R}^d$.*

In this thesis, the orientation of hypersurfaces is an important concept. We will consider i.a. closed curves and surfaces (with no self-intersections) which are orientable. An orientation defines an interior and an exterior of such hypersurfaces. This enables us to consider for example the *area (volume) enclosed by a curve (surface).*

Definition 2.6 (First Fundamental Form)**.** *For vectors $\vec{X}, \vec{Y} \in \mathbb{R}^d$, $\vec{X} \,.\, \vec{Y}$ denotes the Euclidean inner product. For $\vec{p} \in \Gamma$, the Riemannian metric $g = (g_{\vec{p}})_{\vec{p} \in \Gamma}$ given by*

$$g_{\vec{p}} : T_{\vec{p}}\Gamma \times T_{\vec{p}}\Gamma \to \mathbb{R}, \quad (\vec{X}, \vec{Y}) \mapsto \vec{X} \,.\, \vec{Y}, \tag{2.3}$$

is called first fundamental form *of Γ.*

Let $\vec{x} : U \to \mathbb{R}^d$ be a smooth parameterization of Γ. The first fundamental form g can be described by the symmetric and positive definite matrix $(g_{ij})_{i,j=1,\dots,d-1}$ given by

$$g_{ij}(\vec{u}) = \frac{\partial \vec{x}}{\partial u_i}(\vec{u}) \,.\, \frac{\partial \vec{x}}{\partial u_j}(\vec{u}). \tag{2.4}$$

Using a smooth parameterization and the representation (2.4) of the first fundamental form, we can now define integrals on hypersurfaces:

Definition 2.7 (Integral on Hypersurfaces)**.** *Let Γ be a smooth hypersurface with a parameterization $\vec{x} : U \to \mathbb{R}^d$. Let $f : \Gamma \to \mathbb{R}$ be a function defined on the hypersurface Γ. We call f* integrable*, if*

$$\int_U (f \circ \vec{x}) \sqrt{\det(g_{ij})}\, \mathrm{d}u < \infty. \tag{2.5}$$

In this case, we set

$$\int_\Gamma f \, \mathrm{d}A = \int_U (f \circ \vec{x}) \sqrt{\det(g_{ij})}\, \mathrm{d}u. \tag{2.6}$$

We call $\mathrm{d}A$ the area element.

Definition 2.8 (Area of Hypersurfaces)**.** *Let Γ be a smooth, compact hypersurface. The* area *of Γ is defined by*

$$|\Gamma| = \int_\Gamma 1 \, \mathrm{d}A. \tag{2.7}$$

Remark 2.9 (Curves in $\mathbb{R}^2$)**.** *For curves, we often replace U by an interval $I = [a, b] \subset \mathbb{R}$ in the notation, and parameterize a curve $\Gamma \subset \mathbb{R}^2$ by*

$$\vec{x} : I \to \mathbb{R}^2. \tag{2.8}$$

For a smooth, regular parameterized curve, we have $\vec{x}_\rho(\rho) := \frac{\mathrm{d}\vec{x}}{\mathrm{d}\rho}(\rho) \neq \vec{0}$ for each $\rho \in I$, since $\vec{x}$ is an immersion. The vector $\vec{x}_\rho(\rho)$ is a tangent vector on the curve Γ in $\vec{x}(\rho)$. A curve is parameterized by arc-length *if there exist a parameterization $\vec{x} : I \to \mathbb{R}^2$ with $\|\vec{x}_\rho(\rho)\| = 1$ for all $\rho \in I$, where $\|\,.\,\|$ denotes the Euclidean norm. One can show, that each smooth curve can be parameterized by arc-length, see Kühnel (2005).*

The arc-length *is given by*

$$s(\rho_0) = \int_a^{\rho_o} \|\vec{x}_\rho\| \, \mathrm{d}\rho. \tag{2.9}$$

For curves, we often use the notation $\mathrm{d}s$ (length element) instead of $\mathrm{d}A$. The length of the curve is

$$|\Gamma| = \int_\Gamma 1 \, \mathrm{d}s = \int_a^b \|\vec{x}_\rho\| \, \mathrm{d}\rho = s(b). \tag{2.10}$$

To define the second fundamental form, we first consider the change of a function in a tangential direction. The so-called tangential gradient is defined using the concept of directional derivatives:

Definition 2.10 (Directional Derivative)**.** *Let $f : \Gamma \to \mathbb{R}^{(d)}$ be a smooth function or a smooth vector field. For $\vec{p} \in \Gamma$ and $\vec{\tau} \in T_{\vec{p}}\Gamma$, let $\gamma : (-\epsilon, \epsilon) \to \Gamma$, $\epsilon > 0$, be a smooth curve with $\gamma(0) = \vec{p}$ and $\dot{\gamma}(0) = \frac{\mathrm{d}}{\mathrm{d}\epsilon}\gamma(\epsilon)_{|\epsilon=0} = \vec{\tau}$. The derivative of f in direction $\vec{\tau} \in T_{\vec{p}}\Gamma$ is defined as*

$$\partial_{\vec{\tau}} f(\vec{p}) = \frac{\mathrm{d}}{\mathrm{d}\epsilon} f(\gamma(\epsilon))_{|\epsilon=0}. \tag{2.11}$$

Definition 2.11 (Tangential Gradient)**.** *Let $\vec{\tau}_1, \ldots, \vec{\tau}_{d-1}$ be an orthonormal basis of the tangential space $T_{\vec{p}}\Gamma$ for some $\vec{p} \in \Gamma$. Let $f : \Gamma \to \mathbb{R}$ be a smooth function. The* tangential gradient *is defined as*

$$\nabla_\Gamma f(\vec{p}) = \sum_{i=1}^{d-1} (\partial_{\vec{\tau}_i} f(\vec{p})) \vec{\tau}_i. \tag{2.12}$$

Remark 2.12. *(i) We also use the notation* $\nabla_\Gamma f(\vec{p}) = (\underline{D}_1 f(\vec{p}), \ldots, \underline{D}_d f(\vec{p})) \in \mathbb{R}^d$.

(ii) If f *can be extended to an open neighborhood of* Γ *in* $\mathbb{R}^d$*, the surface gradient can be rewritten as*

$$\nabla_\Gamma f = \nabla f - (\nabla f \,.\, \vec{\nu})\vec{\nu}, \tag{2.13}$$

where $\vec{\nu} : \Gamma \to S^{d-1}$ *is a unit normal vector field on* Γ.

Similarly to the concept of tangential gradients, the classical divergence can be generalized to the so-called surface divergence:

Definition 2.13 (Surface Divergence of Vector Fields)**.** *Let* $\vec{\tau}_1, \ldots, \vec{\tau}_{d-1}$ *be an orthonormal basis of the tangential space* $T_{\vec{p}}\Gamma$ *for some* $\vec{p} \in \Gamma$*. Let* $\vec{f} : \Gamma \to \mathbb{R}^d$ *be a smooth vector field. The* surface divergence *of* $\vec{f}$ *in* $\vec{p}$ *is defined as*

$$\nabla_\Gamma \,.\, \vec{f}(\vec{p}) = \sum_{i=1}^{d-1} \vec{\tau}_i \,.\, \partial_{\vec{\tau}_i} \vec{f}(\vec{p}). \tag{2.14}$$

Using the definitions of the tangential gradient and the surface divergence, we can now define the Laplace-Beltrami operator:

Definition 2.14 (Laplace-Beltrami Operator)**.** *For a* C^2*-differentiable function* $f : \Gamma \to \mathbb{R}$*, we define the* Laplace-Beltrami operator *of* f *as*

$$\Delta_\Gamma f(\vec{p}) = \nabla_\Gamma \,.\, \nabla_\Gamma f(\vec{p}), \quad \vec{p} \in \Gamma. \tag{2.15}$$

The Laplace-Beltrami operator applied on functions defined on hypersurfaces is the analogue of the classical Laplace operator applied on functions defined on Euclidean spaces. Similar to the standard Laplace operator, the Laplace-Beltrami operator is often used for smoothing purposes.

Definition 2.15 (Second Fundamental Form)**.** *Let* $\vec{\nu} : \Gamma \to S^{d-1}$ *be a unit normal vector field on* Γ*. The bilinear form*

$$h_{\vec{p}} : T_{\vec{p}}\Gamma \times T_{\vec{p}}\Gamma \to \mathbb{R}, \quad (\vec{\tau}_1, \vec{\tau}_2) \mapsto (-\partial_{\vec{\tau}_1}\vec{\nu}) \,.\, \vec{\tau}_2 \tag{2.16}$$

is called second fundamental form.

Definition 2.16 (Weingarten Map)**.** *The mapping*

$$W_{\vec{p}} : T_{\vec{p}}\Gamma \to T_{\vec{p}}\Gamma, \quad \vec{\tau} \mapsto -\partial_{\vec{\tau}}\vec{\nu} \tag{2.17}$$

is called Weingarten map.

Remark 2.17. *(i) Since* $(-\partial_{\vec{\tau}}\vec{\nu}) \,.\, \vec{\nu} = \partial_\tau(\frac{1}{2}\|\vec{\nu}\|^2) = \partial_\tau(\frac{1}{2}) = 0$*, the Weingarten map is an endomorphism. The linearity is a consequence of the linearity of the operator* $\partial_{\vec{\tau}}$ *w.r.t.* $\vec{\tau}$.

(ii) The Weingarten map is symmetric, i.e.

$$W_{\vec{p}}(\vec{\tau}_1) \,.\, \vec{\tau}_2 = \vec{\tau}_1 \,.\, W_{\vec{p}}(\vec{\tau}_2) \tag{2.18}$$

A proof is given in Eschenburg and Jost (2007) and Eck et al. (2008). As a consequence, real eigenvalues $\kappa_1 \leq \ldots \leq \kappa_{d-1}$ *of the Weingarten mapping exist. The eigenvalues are called* principal curvatures.

(iii) The principal curvatures κ_1 and κ_{d-1} minimize and maximize the second fundamental form.

Definition 2.18 (Mean Curvature)**.** *For $\vec{p} \in \Gamma$, the sum*

$$\kappa(\vec{p}) = \operatorname{trace}(W_{\vec{p}}) = \sum_{i=1}^{d-1} \kappa_i \tag{2.19}$$

is called the mean curvature *of Γ in $\vec{p}$.*

Using the surface divergence, the mean curvature can also be expressed as

$$\kappa = -\nabla_\Gamma \,.\, \vec{\nu}. \tag{2.20}$$

The mean curvature thus describes the change of the normal vector field $\vec{\nu}$ which is the intuitive description of curvature.

Lemma 2.19. *Let $\vec{x} : U \to \mathbb{R}^d$ be a smooth (at least C^2-) parameterization of Γ and $\vec{\nu} : \Gamma \to S^{d-1}$ a normal vector field. Then, the mean curvature κ and the Laplace-Beltrami operator applied on $\vec{x}$ are related by*

$$\Delta_\Gamma \vec{x} = \kappa \vec{\nu}. \tag{2.21}$$

Proof. See e.g. Deckelnick et al. (2005). □

Equation 2.21 is frequently used in this thesis to relate curvature κ and parameterization $\vec{x}$. In the next chapters of this thesis, we will consider a variety of evolution equations for curves and surfaces which contain the mean curvature for smoothing purposes. Such evolution equations consist of a forcing term designed for the special application, for example, designed to detect edges in given images. In addition, a curvature term is used which provides smoothness of the evolving hypersurface.

Remark 2.20. *(i) Note, that the sign of the mean curvature κ depends on the choice of $\vec{\nu}$. Since we require $\vec{\nu}$ to be a unit normal vector field, i.e. $\|\vec{\nu}(\vec{p})\| = 1$ for all $\vec{p} \in \Gamma$, there are two possible choices for $\vec{\nu}$. However, the* mean curvature vector

$$\vec{\kappa} := \kappa \vec{\nu} \tag{2.22}$$

is independent on the choice of $\vec{\nu}$.

(ii) For a curve $\Gamma \in \mathbb{R}^2$, the Laplace-Beltrami operator applied on a function is identical to the second derivative of the function with respect to arc-length, i.e. $\Delta_\Gamma f = f_{ss} = \frac{\partial^2}{\partial s^2} f$. Note, that $\frac{\partial}{\partial s} = \frac{1}{\|\vec{x}_\rho\|} \frac{\partial}{\partial \rho}$. In particular, it follows that $\vec{x}_{ss} = \kappa \vec{\nu}$.

Remark 2.21 (Hypersurfaces with Boundary)**.** *A hypersurface with non-empty boundary is a $(d-1)$ dimensional submanifold Γ with a topological boundary $\partial\Gamma \neq \emptyset$ which is a $(d-2)$ dimensional submanifold of $\mathbb{R}^d$. In detail, for $\vec{p} \in \partial\Gamma$ there exist a point $\vec{u}_0 = (u_{0,1}, \ldots, u_{0,d-2}, 0) \in \mathbb{R}^{d-1}$, an open neighborhood U of $\vec{u}_0$ in $\mathbb{R}^{d-1}$ and an immersion $\vec{x} : U \to \mathbb{R}^d$ with $\vec{x}(\vec{u}_0) = \vec{p}$, $\vec{x}(U \cap \mathbb{R}^{d-1,+}) \subset \Gamma$ and $\vec{x}(U \cap (\mathbb{R}^{d-2} \times \{0\})) \subset \partial\Gamma$. Here, $\mathbb{R}^{d-1,+}$ is the set $\mathbb{R}^{d-1,+} = \{\vec{u} = (u_1, \ldots, u_{d-1}) \in \mathbb{R}^{d-1} \,:\, u_{d-1} > 0\}$.*

For each $\vec{p} \in \partial\Gamma$, we can define a co-normal *$\vec{\mu} \in T_{\vec{p}}\Gamma$, which is orthogonal to the elements in $T_{\vec{p}}\partial\Gamma$. The vector $\vec{\mu}$ is an* outer *co-normal if a curve $\gamma : [-\epsilon, 0] \to \mathbb{R}^d$ exists, with*

$\gamma([-\epsilon,0]) \subset \Gamma$, $\gamma(0) = \vec{p} \in \partial\Gamma$ *and* $\dot{\gamma}(0) = \vec{\mu}$. *Using the parameterization* $\vec{x}$ *as above, an outer co-normal is given by* $\vec{\mu} = -\frac{\partial \vec{x}}{\partial u_{d-1}}(\vec{u}_0)$.

Integration on $\partial\Gamma$ *is defined similar as on* Γ *(recall Definition 2.7), by using an immersion* $\vec{x}_{\partial\Gamma} : V \to \mathbb{R}^d$, $V \subset \mathbb{R}^{d-2}$, *with* $\vec{x}_{\partial\Gamma}(V) = \partial\Gamma$, *i.e.*

$$\int_{\partial\Gamma} f \,\mathrm{d}r := \int_V (f \circ \vec{x}_{\partial\Gamma}) \sqrt{\det(\tilde{g}_{ij})} \,\mathrm{d}v. \tag{2.23}$$

for a function $f : \partial\Gamma \to \mathbb{R}$. *Here,* $\tilde{g}_{ij}(\vec{v}) := \frac{\partial \vec{x}_{\partial\Gamma}}{\partial v_i}(\vec{v}) \,.\, \frac{\partial \vec{x}_{\partial\Gamma}}{\partial v_j}(\vec{v})$, $i, j = 1, \ldots, d-2$.

Using the concept of *hypersurfaces with boundary*, we can formulate an *integration by parts* theorem. This is a basic theorem frequently used when applying techniques of the theory of calculus of variations; for example when deriving necessary conditions from a minimization problem.

Theorem 2.22 (Integration by parts, Green's formula)**.** *Let* $\Gamma \subset \mathbb{R}^d$ *be a smooth, bounded hypersurface with a possible non-empty smooth boundary* $\partial\Gamma$. *Let* $\vec{\nu} = (\nu_1, \ldots, \nu_d) : \Gamma \to S^{d-1}$ *denote a unit normal vector field on* Γ. *If* $\partial\Gamma \neq \emptyset$, *let* $\vec{\mu} = (\mu_1, \ldots, \mu_d) : \partial\Gamma \to S^{d-1}$ *be an outer unit co-normal vector field at* $\partial\Gamma$. *Then, for smooth functions* $f, g : \Gamma \to \mathbb{R}$ *we have*

$$\int_\Gamma \underline{D}_i f \,\mathrm{d}A = \int_\Gamma f \kappa \nu_i \,\mathrm{d}A + \int_{\partial\Gamma} f \mu_i \,\mathrm{d}r, \quad i = 1, \ldots, d. \tag{2.24}$$

and

$$\int_\Gamma \nabla_\Gamma f \,.\, \nabla_\Gamma g \,\mathrm{d}A = -\int_\Gamma f \Delta_\Gamma g \,\mathrm{d}A + \int_{\partial\Gamma} f \nabla_\Gamma g \,.\, \vec{\mu} \,\mathrm{d}r. \tag{2.25}$$

Proof. See Dziuk and Elliott (2013). □

2.1.2 Representation of Hypersurfaces

There are several possibilities to represent a hypersurface, see Deckelnick et al. (2005), Giga (2006). There exist two main classes how a hypersurface can be represented: direct methods like parametric and graph-based methods and indirect methods like level set and phase field approaches. In this section, we will shortly introduce these four different ways to represent a hypersurface.

Parameterized Hypersurfaces: In Section 2.1.1, we used a parametric approach $\vec{x} : U \to \mathbb{R}^d$ to describe a hypersurface, where $U \subset \mathbb{R}^{d-1}$ was an open subset. Alternatively, a hypersurface Γ can be parameterized by $\vec{x} : \mathcal{M} \to \mathbb{R}^d$, where $\mathcal{M} \subset \mathbb{R}^d$ is a suitable reference manifold of the same topology type. For example, a closed curve $\Gamma \subset \mathbb{R}^2$ can be parameterized over the unit circle $\mathcal{M} = S^1 \subset \mathbb{R}^2$. The curvature of a parameterized surface is related to $\vec{x}$ via (2.21).

Graph Hypersurfaces: Let $\Omega \subset \mathbb{R}^{d-1}$ be an open subset and $h : \Omega \to \mathbb{R}$ a C^k-function, $k \in \mathbb{N}$. Then, the graph of h

$$\Gamma = \{(\vec{z}, h(\vec{z})) \,:\, \vec{z} \in \Omega\} \tag{2.26}$$

is a C^k-hypersurface. Note, that a parameterization of Γ is given by $\vec{x} = (\vec{\mathrm{Id}}_{d-1}|_\Omega, h) : \Omega \to \mathbb{R}^d$, where $\vec{\mathrm{Id}}_{d-1} : \mathbb{R}^{d-1} \to \mathbb{R}^{d-1}$ is the identity mapping of $\mathbb{R}^{d-1}$. Thus,

$$\vec{\tau}_1 = \left(1, 0, \ldots, 0, \frac{\partial h(\vec{z})}{\partial \vec{z}_1}\right), \ldots, \vec{\tau}_{d-1} = \left(0, \ldots, 0, 1, \frac{\partial h(\vec{z})}{\partial \vec{z}_{d-1}}\right) \tag{2.27}$$

are a basis of $T_{\vec{p}}\Gamma$ for $\vec{p} = (\vec{z}, h(\vec{z}))$. Consequently, a normal vector is given by

$$\vec{\nu} = \pm \frac{(\nabla_{\vec{z}} h, -1)}{\sqrt{1 + \|\nabla_{\vec{z}} h\|^2}} \tag{2.28}$$

and for $k \geq 2$ the curvature of Γ can be expressed as

$$\kappa = \mp \nabla_{\vec{z}} \cdot \left(\frac{\nabla_{\vec{z}} h}{\sqrt{1 + \|\nabla_{\vec{z}} h\|^2}} \right), \tag{2.29}$$

see Eck et al. (2008). Here, $\nabla_{\vec{z}}$ denotes the gradient w.r.t. $\vec{z} \in \mathbb{R}^{d-1}$.

Level Set Approach A C^k-hypersurface Γ can also be locally represented by the zero level set of a C^k-function. In detail, for each point $\vec{p} \in \Gamma$ there exists a neighborhood $U \subset \mathbb{R}^d$ of $\vec{p}$ in $\mathbb{R}^d$ and a C^k-function $u : U \to \mathbb{R}$, with $\nabla u(\vec{x}) \neq 0$ for all $\vec{x} \in \Gamma \cap U$ and

$$\Gamma \cap U = \{\vec{x} \in U \,:\, u(\vec{x}) = 0\}. \tag{2.30}$$

The one-dimensional normal space $N_{\vec{x}}\Gamma$ for $\vec{x} \in \Gamma$ is spanned by

$$\vec{\nu}(\vec{x}) = \pm \frac{\nabla u(\vec{x})}{\|\nabla u(\vec{x})\|}, \tag{2.31}$$

and $T_{\vec{x}}\Gamma = (N_{\vec{x}}\Gamma)^{\perp}$. Using (2.20), it can be shown that

$$\kappa = \mp \nabla_{\Gamma} \cdot \left(\frac{\nabla u(\vec{x})}{\|\nabla u(\vec{x})\|} \right) = \mp \nabla \cdot \left(\frac{\nabla u(\vec{x})}{\|\nabla u(\vec{x})\|} \right) \tag{2.32}$$

for $k \geq 2$. The level set approach is thus an indirect method to describe a hypersurface. It is a very popular approach to describe time-varying hypersurfaces (see Section 2.1.3 and Deckelnick et al. (2005)), since it allows for topology changes whereas the topology of a parametric or graph hypersurface is fixed.

Phase Field Approach Another indirect method is the phase field approach. We assume that Γ is an interface between two disjoint bulks $\Omega_1, \Omega_2 \subset \mathbb{R}^d$. The *sharp* interface Γ is approximated by a so-called *diffuse* interface

$$\Gamma^{\epsilon} = \left\{ \vec{x} \in \mathbb{R}^d \,:\, -1 + C\epsilon \leq u_{\epsilon}(\vec{x}) \leq 1 - C\epsilon \right\}, \tag{2.33}$$

for some $C > 0$. Here, u_{ϵ} is a phase field function, i.e. its value is -1 in a bulk Ω_1^{ϵ} and 1 in a bulk Ω_2^{ϵ}. The zero level set of the phase field function is an approximation of the sharp interface Γ. The phase field approach also allows for topology changes. For more details on the phase field approach, we refer to Deckelnick et al. (2005).

In this thesis, we will follow a parametric approach to describe hypersurfaces, but we will also compare parametric methods with level set methods. Direct methods can be used to efficiently represent complex networks like, for example, triple junctions in the case of curves (i.e. $d = 2$). A curve network with triple junctions cannot be represented by one single level set or phase field function, see also Section 3.2.4. We do not follow a graph based approach since many hypersurfaces cannot be represented by a graph. However, we note that a local description of a hypersurface by a graph can be very useful for local considerations.

Indirect methods like the level set or phase field approach allow for automatic topology changes. However, in this thesis, we will use and develop efficient methods to detect and perform topology changes involving parametric curves and surfaces. Being able to handle topology changes, the parametric method can be used for a wide range of applications including image segmentation applications.

2.1.3 Evolving Hypersurfaces and Transport Theorem

In this section we consider time-varying hypersurfaces, called *evolving* hypersurfaces.

Definition 2.23 (Smooth Evolving Hypersurface)**.** *A set $\Gamma \subset \mathbb{R}^{d+1}$ is called* smooth, evolving hypersurface *of $\mathbb{R}^d$, if there exist a $T > 0$, such that*

(i) *Γ is a smooth hypersurface of $\mathbb{R}^d \times \mathbb{R}$,*

(ii) *Γ can be expressed as*

$$\Gamma = \{(\vec{x}, t) \,:\, t \in (0,T),\, \vec{x} \in \Gamma(t)\}\,, \tag{2.34}$$

with smooth hypersurfaces $\Gamma(t)$ of $\mathbb{R}^d$,

(iii) *the tangential spaces $T_{(\vec{x},t)}\Gamma$ satisfy*

$$T_{(\vec{x},t)}\Gamma \neq \mathbb{R}^d \times \{0\},\ \forall (\vec{x}, t) \in \Gamma. \tag{2.35}$$

The motion of evolving hypersurfaces can be expressed by its so-called *normal velocity*, which describes the change of the hypersurface in normal direction.

Definition 2.24 (Normal velocity)**.** *For $t \in (0,T)$ let $\vec{x}(\,.\,,t) : U \to \mathbb{R}$ be a parameterization of $\Gamma(t)$ and let $\vec{\nu}(\,.\,,t) : \Gamma(t) \to S^{d-1}$ be a normal vector field on $\Gamma(t)$. Further let the mapping $(\vec{u}, t) \mapsto \vec{x}(\vec{u}, t)$ be smooth in the variable t and let $\vec{x}_t$ denote the partial derivative of that mapping with respect to t.*

The normal velocity of Γ is defined by

$$V_n = \vec{x}_t \,.\, \vec{\nu}. \tag{2.36}$$

It is worth mentioning that the normal velocity completely describes the motion of a hypersurface. The change of $\vec{x}$ in a tangential direction $\vec{\tau}$, i.e. $\vec{x}_t \,.\, \vec{\tau}$, induces only a reparameterization of the hypersurface, i.e. a hypersurface $\Gamma(t)$ will not change if $V_n = 0$ but $\vec{x}_t \,.\, \vec{\tau} \neq 0$.

Now we formulate a basic theorem, the transport theorem. It is used to compute the time-derivative of $\int_{\Gamma(t)} f \,\mathrm{d}A$, where $\Gamma(t)$ is an time-dependent hypersurface and f is a function with an extension to some small bulk around the evolving hypersurface. Note, that both $\Gamma(t)$ and f are time-dependent. Thus when considering the time-derivative of $\int_{\Gamma(t)} f \,\mathrm{d}A$, we will obtain one term containing the partial derivative of f with respect to the time and terms containing the curvature and normal velocity of the evolving hypersurface.

Theorem 2.25 (Transport Theorem)**.** *Let $\Omega \subset \mathbb{R}^d$ be an open subset, and Γ a smooth evolving hypersurface with $\Gamma(t) \subset\subset \Omega$ for all $t \in (0,T)$. Further, we assume that $\Gamma(t)$ is the boundary of an open subset $\Omega_1(t) \subset \Omega$. Let $\Omega_2(t) = \Omega \setminus \overline{\Omega_1(t)}$.*

Let $\Omega_i := \{(\vec{x}, t) \,:\, t \in (0,T),\, \vec{x} \in \Omega_i(t)\}$. We choose a normal vector field $\vec{\nu}(\,.\,,t)$ on $\Gamma(t)$ such that $\vec{\nu}(\,.\,,t)$ points from $\Omega_2(t)$ to $\Omega_1(t)$.

Let $f : \Omega \times (0,T) \to \mathbb{R}$ be a function, such that $f_{|\Omega_i}$ has a C^1-extension to $\overline{\Omega_i}$ for $i = 1, 2$. For $\vec{x} \in \Gamma(t)$, we set $f_i(\vec{x}, t) := \lim_{\vec{y} \to \vec{x}, \vec{y} \in \Omega_i(t)} f(\vec{y}, t)$. Then, the following formulas hold:

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{\Omega_i(t)} f \,\mathrm{d}x = \int_{\Omega_i(t)} \frac{\partial f}{\partial t} \,\mathrm{d}x + (-1)^i \int_{\Gamma(t)} f_i V_n \,\mathrm{d}A, \tag{2.37}$$

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_{\Omega} f \,\mathrm{d}x = \int_{\Omega_1(t)} \frac{\partial f}{\partial t} \,\mathrm{d}x + \int_{\Omega_2(t)} \frac{\partial f}{\partial t} \,\mathrm{d}x - \int_{\Gamma(t)} (f_1 - f_2) V_n \,\mathrm{d}A. \tag{2.38}$$

If $g \in C^1(Q_\Gamma)$ for an open set $Q_\Gamma \subset \mathbb{R}^{d+1}$ containing $\bigcup_{t\in(0,T)} \Gamma(t) \times \{t\}$, we have

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Gamma(t)} g \,\mathrm{d}A = \int_{\Gamma(t)} \frac{\partial g}{\partial t}\,\mathrm{d}A + \int_{\Gamma(t)} \frac{\partial g}{\partial \nu} V_n \,\mathrm{d}A - \int_{\Gamma(t)} g\kappa V_n \,\mathrm{d}A. \tag{2.39}$$

Proof. See Deckelnick et al. (2005) and Eck et al. (2008). □

Note that in Theorem 2.25, we considered hypersurfaces with $\partial\Gamma(t) = \emptyset$, since $\Gamma(t)$ was the boundary of an open set $\Omega_1(t)$. In Chapter 3, we will also consider curves with non-empty boundary. In such cases, we will have additional terms in equations like (2.39) containing the tangential velocity of the boundary points.

Remark 2.26 (Mean Curvature Flow and Related Flows)**.** *In the following chapters of this thesis, we consider flows of hypersurfaces of the form*

$$V_n = f\kappa + F, \tag{2.40}$$

with functions $f, F : \Gamma \to \mathbb{R}$. The function f is a weighting function for the curvature term and F is called external forcing term. In case $f \equiv 1$ and $F \equiv 0$, the flow reduces to the well-known mean curvature flow

$$V_n = \kappa. \tag{2.41}$$

For $d = 2$, the flow is also known as curve shortening flow*.*

From (2.39)*, we obtain*

$$\frac{\mathrm{d}}{\mathrm{d}t}\,|\Gamma(t)| = -\int_{\Gamma(t)} \kappa\, V_n \,\mathrm{d}A. \tag{2.42}$$

Thus, the mean curvature flow (2.41) *decreases the surface area. In fact, the mean curvature flow is the gradient flow of the area functional*

$$E(\Gamma) = |\Gamma| = \int_\Gamma 1 \,\mathrm{d}A, \tag{2.43}$$

see e.g. Garcke (2013).

For properties of the mean curvature flow, we refer to the classical works of Huisken (1984), Gage and Hamilton (1986) and Grayson (1987). We further refer to Ilmanen (1998), Deckelnick et al. (2005), Giga (2006) and Garcke (2013) and the references therein for geometric surface evolution equations, including mean curvature flow and related flows.

For image segmentation purposes, we will consider flows of the form (2.40)*. These flows can be derived as gradient flows of corresponding energy functionals. Typical energy functionals for image processing tasks contain weighted area terms similar to* (2.43)*, which are denoted as internal energies. Further, they contain external energy terms, which are specially designed for e.g. edge detection or image segmentation.*

2.2 Curves on Surfaces

In the previous section, we considered hypersurfaces, i.e. $(d-1)$-dimensional submanifolds of $\mathbb{R}^d$. In this thesis, we further consider curves in $\mathbb{R}^3$ which are restricted to lie on two-dimensional surfaces.

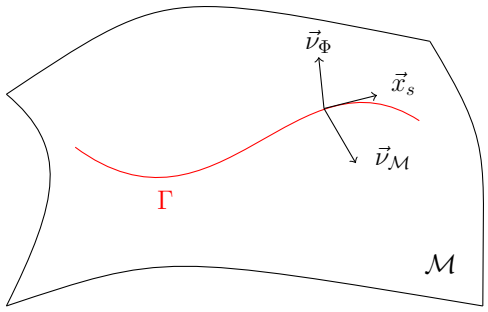


Figure 2.1: Illustration of the vector fields $\vec{x}_s$, $\vec{\nu}_\Phi$ and $\vec{\nu}_\mathcal{M} = \vec{x}_s \times \vec{\nu}_\Phi$.

Let $\mathcal{M} \subset \mathbb{R}^3$ be a smooth two-dimensional manifold. Locally, $\mathcal{M}$ can be described as the zero level set of a smooth function defined in $\mathbb{R}^3$, recall (2.30). Using a partition of unity, we therefore assume that

$$\mathcal{M} = \left\{ \vec{z} \in \mathbb{R}^3 \,:\, \Phi(\vec{z}) = 0 \right\}, \tag{2.44}$$

for a function $\Phi \in C^2(\mathbb{R}^3, \mathbb{R})$ with $\|\nabla\Phi(\vec{z})\| > 0$ for $\vec{z} \in \mathcal{M}$. A unit normal vector field $\vec{n}_\Phi$ on $\mathcal{M}$ is given by

$$\vec{n}_\Phi(\vec{z}) := \frac{\nabla\Phi(\vec{z})}{\|\nabla\Phi(\vec{z})\|}$$

for $\vec{z} \in \mathcal{M}$.

Definition 2.27. *Let $\Gamma \subset \mathcal{M}$ be a curve on a smooth surface $\mathcal{M} \subset \mathbb{R}^3$. Let Γ be parameterized by $\vec{x} : I \to \mathcal{M}$, where I is a one-dimensional reference manifold, e.g. the unit interval $I = [0, 1]$ for open curves or the unit circle $I = S^1$ for closed curves. We define $\vec{\nu}_\Phi : I \to \mathbb{R}^3$ such that*

$$\vec{\nu}_\Phi(\rho) := \vec{n}_\Phi(\vec{x}(\rho))$$

is the surface normal evaluated at $\vec{x}(\rho)$ for $\rho \in I$. Further, we define $\vec{\nu}_\mathcal{M} : I \to \mathbb{R}^3$ by

$$\vec{\nu}_\mathcal{M}(\rho) := \vec{x}_s(\rho) \times \vec{\nu}_\Phi(\rho).$$

We see that $\vec{\nu}_\mathcal{M}$ is perpendicular to $\vec{x}_s$, i.e. normal to the curve, but lies in the tangent space to $\mathcal{M}$. Figure 2.1 illustrates a possible surface $\mathcal{M}$, a curve Γ and the vector fields $\vec{\nu}_\Phi$, $\vec{x}_s$ and $\vec{\nu}_\mathcal{M}$.

Further, the vector $\vec{x}_{ss}$ which is perpendicular to $\vec{x}_s$ can be written as the sum of its component in $\vec{\nu}_\Phi$-direction and its component in $\vec{\nu}_\mathcal{M}$-direction. This motivates the following definition, cf. Barrett et al. (2010a):

Definition 2.28. *We define the* geodesic curvature $\kappa_\mathcal{M} : I \to \mathbb{R}$ *and the* normal curvature $\kappa_\Phi : I \to \mathbb{R}$ *by*

$$\kappa_\mathcal{M} = \vec{x}_{ss} \,.\, \vec{\nu}_\mathcal{M}, \quad \kappa_\Phi = \vec{x}_{ss} \,.\, \vec{\nu}_\Phi. \tag{2.45}$$

As a consequence, $\vec{x}_{ss}$ can be expressed as

$$\vec{x}_{ss} = \kappa_\mathcal{M}\vec{\nu}_\mathcal{M} + \kappa_\Phi\vec{\nu}_\Phi. \tag{2.46}$$

Remark 2.29. *Let $f : \mathcal{M} \to \mathbb{R}$ be a smooth function defined on the surface $\mathcal{M}$. Let $\vec{y} : I \times (-\epsilon_0, \epsilon_0) \to \mathcal{M}$ be a family of parameterizations of curves on the surface. Consider*

$\gamma(\epsilon) = \vec{y}(\rho, \epsilon)$ *for a fixed* ρ*. From the definition of the directional derivative* (2.11) *and of the surface gradient* (2.12)*, we get the following chain rule on manifolds:*

$$\frac{\mathrm{d}}{\mathrm{d}\epsilon} f(\vec{y}(\rho,\epsilon))|_{\epsilon=0} = \nabla_{\mathcal{M}} f(\vec{y}(\rho,0)) \,.\, \vec{y}_\epsilon(\rho,0). \tag{2.47}$$

In the previous section, we have defined integrals on hypersurfaces, see Definition 2.7. We can also define integrals on sub-manifolds of higher co-dimension. Here, we consider in particular the case of curves in $\mathbb{R}^3$:

Definition 2.30 (Integral over Curves in $\mathbb{R}^3$)**.** *Let* $\Gamma \subset \mathbb{R}^3$ *be a smooth curve. Let* $\vec{x} : I \to \mathbb{R}^3$ *be a parameterization of* Γ*. A function* $f : \Gamma \to \mathbb{R}$ *is called integrable if*

$$\int_I f(\vec{x}(\rho)) \|\vec{x}_\rho(\rho)\| \,\mathrm{d}\rho < \infty. \tag{2.48}$$

If f *is integrable, we define*

$$\int_\Gamma f \,\mathrm{d}s = \int_I f(\vec{x}(\rho)) \|\vec{x}_\rho(\rho)\| \,\mathrm{d}\rho. \tag{2.49}$$

We now consider the equivalent of mean curvature flows of hypersurfaces (see Remark 2.26) for curves on surfaces:

Remark 2.31 (Geodesic Curvature Flow and Related Flows)**.** *Analogue to the length functional for planar curves, we can consider the length functional for curves on surfaces*

$$E(\Gamma) = |\Gamma| = \int_\Gamma 1 \,\mathrm{d}s, \tag{2.50}$$

Barrett et al. (2010a) derived the geodesic curvature flow

$$\vec{x}_t \,.\, \vec{\nu}_{\mathcal{M}} = \kappa_{\mathcal{M}} \tag{2.51}$$

as a gradient flow of the length functional (2.50) *using methods from the theory of calculus of variations. For that, the variations are restricted to lie on the surface* $\mathcal{M}$*.*

Stationary solutions of the geodesic curvature flow are curves that minimize the length functional. They are called geodesics*.*

For image segmentation, we will consider variants of this flow, containing the curvature $\kappa_{\mathcal{M}}$*, weighted with some function or constant, plus an external forcing term:*

$$\vec{x}_t \,.\, \vec{\nu}_{\mathcal{M}} = f\kappa_{\mathcal{M}} + F. \tag{2.52}$$

Definition 2.32 (Geodesic distance)**.** *For* $\vec{p}_1, \vec{p}_2 \in \mathcal{M}$*, let* Γ *be a curve on* $\mathcal{M}$ *which minimizes* (2.50) *among all curves which connect* $\vec{p}_1$ *and* $\vec{p}_2$*. The length* $|\Gamma|$ *is called* geodesic distance *between* $\vec{p}_1$ *and* $\vec{p}_2$*.*

Chapter 3

Two-dimensional Image Processing

This chapter focuses on the processing of planar, two-dimensional images. Methods for image segmentation and image denoising are presented which are based on partial differential equations. The equations are derived by minimizing an energy functional and by using methods from the theory of calculus of variations. Models for image segmentation and denoising are developed and analyzed. Further, numerical approximations based on finite elements and finite differences are developed and results are presented where the methods are applied on artificial and real, gray-scaled and colored images.

3.1 Introduction

Mathematically, a planar image is given by an image function u_0 defined on a rectangular image domain $\Omega \subset \mathbb{R}^2$. The image function can be scalar-valued or vector-valued.

An example for a scalar-valued image function is a gray-scaled image defined by $u_0 : \Omega \to [0, 1]$, where 0 and 1 correspond to black and white. A color image is given by a vector-valued image function, for example by an RGB image function $u_0 = (u_{0,r}, u_{0,g}, u_{0,b}) : \Omega \to [0, 1]^3$, where the components $u_{0,r}$, $u_{0,g}$ and $u_{0,b}$ describe the intensity of the red, green and blue part of the color. For certain other color spaces, the values of u_0 can also lie on a submanifold of $\mathbb{R}^n$ for some $n \in \mathbb{N}$, see Section 3.2.9.

A real image is generated by an optical sensor, for example by a charge-coupled device (CCD) or a complementary metal oxide semiconductor (CMOS) camera. Camera images are raster images where a brightness or color is given for a rectangular ordered set of so-called pixels. Let an image with $N_x \times N_y$ pixels be given. The rectangular image domain Ω is thus decomposed into $N_x \times N_y$ squares representing the pixels. The image function u_0 is piecewise constant and defined for each pixel.

Image segmentation divides an image in characteristic regions or phases, i.e. in subsets of Ω. The regions can represent several objects in the image or the background of the image, cf. Chapter 1. The regions are separated by sharp interfaces or boundaries. The boundaries can be edges with high gradient of the image intensity function. But also so-called weak edges with smooth transitions between image intensity values can occur (Chan and Vese, 2001). In this thesis, image segmentation is performed by making use of so-called Active Contours methods which were originally developed by Kass et al. (1988). There, smooth curves, called contours, evolve in time such that a certain energy functional is minimized. In this work, we mainly focus on active contours for minimizing the functional of Mumford and Shah (1989).

This functional does also involve an approximation u of u_0. Evolution equations for the contours as well as a second order, elliptic partial differential equation for u can be obtained by considering the Mumford-Shah functional, see also Aubert and Kornprobst (2006). The solution u acts as denoised, smooth image approximation.

We perform a two step approach: First, we perform a segmentation of a given image by using curve evolutions. Having detected the regions, the image is smoothed only in the detected, homogeneous regions. Therefore, a blurring of edges can be prevented since a diffusion equation with Neumann boundary conditions is solved separately in each region.

3.2 Methods for Image Segmentation and Restoration

3.2.1 Overview on Active Contours Models

Let $\Gamma \subset \Omega$ be a smooth curve. Many active contours models involve an energy of the form

$$E(\Gamma) = E_{\text{int}}(\Gamma) + E_{\text{ext}}(\Gamma) \tag{3.1}$$

which should be minimized. The internal energy $E_{\text{int}}(\Gamma)$ can contain length and area terms. The external energy $E_{\text{ext}}(\Gamma)$ is designed such that it is small near edges or region boundaries and larger in homogeneous regions.

Minimizing the energy (3.1) leads to evolution equations for time-dependent contours $\Gamma(t)$, also called snakes (Kass et al., 1988). Figure 3.1 illustrates the basic idea of the snake method.

Typically, evolution equations are of the form

$$\text{normal velocity } = \text{curvature term } + \text{ external force.} \tag{3.2}$$

The curvature weighted with some constant or with a function controls the smoothness of the curve. The external force pushes the contour to the desired edges or object boundaries.

Active contours methods can be divided in two main classes: edge-based and region-based active contours. In the following, we first consider scalar, gray-scaled images. Later we state how the methods can be extended to color images.

3.2.2 Edge-based Active Contours

Edge-based active contours make use of the gradient ∇u to indicate an edge, where $u : \Omega \to \mathbb{R}$ is a smoothed version of u_0. For example, u can be set to the convolution of u_0 with a Gaussian smoothing kernel. The energy is designed such that it is small if $\|\nabla u\|$ evaluated along a closed curve Γ is big.

The classical active contours model of Kass et al. (1988) aims at minimizing an energy of the form (3.1) with an internal and external energy given by

$$E_{\text{int}}(\Gamma) = \frac{1}{2}\,\alpha \int_I \|\vec{x}_\rho(\rho)\|^2 \,\mathrm{d}\rho + \frac{1}{2}\,\beta \int_I \|\vec{x}_{\rho\rho}(\rho)\|^2 \,\mathrm{d}\rho, \tag{3.3a}$$

$$E_{\text{ext}}(\Gamma) = -\frac{1}{2}\,\lambda \int_I \|\nabla u(\vec{x}(\rho))\|^2 \,\mathrm{d}\rho, \tag{3.3b}$$

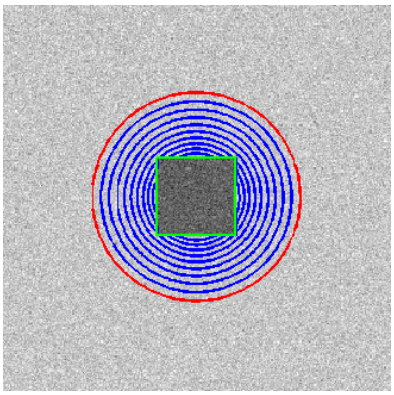


Figure 3.1: Illustration of the active contours/snake method to detect an object in a given image. Starting with an initial contour (red) the curve evolves to the final contour (green) which matches with the object's boundary. The intermediate curves are drawn in blue color.

where α, β, $\lambda > 0$ are weighting parameters and $\vec{x} : I \to \mathbb{R}^2$ is a smooth parameterization of Γ with I being a one-dimensional reference manifold such as the sphere $S^1 \subset \mathbb{R}^2$. More generally, the external energy can be of the form

$$E_{\text{ext}}(\Gamma) = \lambda \int_I g\left(\|\nabla u(\vec{x}(\rho))\|\right) \mathrm{d}\rho, \tag{3.4}$$

with a strictly decreasing function $g : [0, \infty) \to \mathbb{R}$ as proposed by Perona and Malik (1990) for edge detection. For instance, the function g can be defined as $g(p) = -\frac{1}{2}p^2$ as in (3.3b) or $g(p) = 1/(1+p^2)$, where the latter even provides the external energy to be bounded from below. Let the edge indicator function $f : \Omega \to \mathbb{R}$ be defined by $f(\vec{z}) := g\left(\|\nabla u(\vec{z})\|\right)$.

Let Γ be a closed curve. For all $\vec{\eta} : I \to \mathbb{R}^2$, we obtain the first variation of $E(\Gamma)$ in the direction $\vec{\eta}$ as follows, cf. Barrett et al. (2010a): Choose $\vec{y} : I \times (-\epsilon_0, \epsilon_0) \to \mathbb{R}^2$ with $\vec{y}(\rho, 0) = \vec{x}(\rho)$ and $\vec{y}_\epsilon(\rho, 0) = \vec{\eta}(\rho)$. For example, we can use $\vec{y}(\rho, \epsilon) := \vec{x}(\rho) + \epsilon\,\vec{\eta}(\rho)$. We compute

$$\begin{aligned}
\left(\delta E(\Gamma)\right)(\vec{\eta}) &:= \frac{\mathrm{d}}{\mathrm{d}\epsilon}\bigg|_{\epsilon=0} \left(\frac{1}{2}\,\alpha \int_I \|\vec{y}_\rho\|^2 \,\mathrm{d}\rho + \frac{1}{2}\,\beta \int_I \|\vec{y}_{\rho\rho}\|^2 \,\mathrm{d}\rho + \lambda \int_I f(\vec{y})\,\mathrm{d}\rho\right) \\
&= \left(\alpha \int_I \vec{y}_\rho \,.\, \vec{y}_{\rho\epsilon}\,\mathrm{d}\rho + \beta \int_I \vec{y}_{\rho\rho} \,.\, \vec{y}_{\rho\rho\epsilon}\,\mathrm{d}\rho + \lambda \int_I \nabla f(\vec{y}) \,.\, \vec{y}_\epsilon\,\mathrm{d}\rho\right)\bigg|_{\epsilon=0} \\
&= \alpha \int_I \vec{x}_\rho \,.\, \vec{\eta}_\rho\,\mathrm{d}\rho + \beta \int_I \vec{x}_{\rho\rho} \,.\, \vec{\eta}_{\rho\rho}\,\mathrm{d}\rho + \lambda \int_I \nabla f(\vec{x}) \,.\, \vec{\eta}\,\mathrm{d}\rho \\
&= \int_I \left(-\alpha\,\vec{x}_{\rho\rho} + \beta\,\vec{x}_{\rho\rho\rho\rho} + \lambda \nabla f(\vec{x})\right) \,.\, \vec{\eta}\,\mathrm{d}\rho,
\end{aligned} \tag{3.5}$$

where the last identity is obtained by using integration by parts on noting that $\partial\Gamma = \emptyset$ (cf. Theorem 2.22).

In order to define a gradient flow for the energy $E(\Gamma)$, we use an inner product for functions $\vec{\eta} : I \to \mathbb{R}^2$ as proposed by Barrett et al. (2010a). It is worth mentioning, that the shape of a time-dependent curve $\Gamma(t)$ depends only on the normal velocity of its parameterization. The tangential velocity corresponds only to reparameterizations of the curve. Thus, two different parameterizations of a curve with equal normal velocity describe the same evolution, even if they differ in their tangential velocity. Motivated by this fact, we define as in Barrett et al. (2010a) an inner product for functions $\vec{\eta}, \vec{\chi} : I \to \mathbb{R}^2$ by

$$(\vec{\eta}, \vec{\chi})_{2,nor} := \int_\Gamma \vec{P}_n\,\vec{\eta} \,.\, \vec{P}_n\,\vec{\chi}\,\mathrm{d}s, \tag{3.6}$$

where s is the arc-length and $\vec{P}_n := \vec{\mathrm{Id}}_2 - \vec{x}_s \otimes \vec{x}_s$ is the projection onto the normal part. Here, $\vec{\mathrm{Id}}_2$ is the identity operator on $\mathbb{R}^2$. We note that, $\vec{P}_n^T = \vec{P}_n$, $\vec{P}_n^2 = \vec{P}_n$. In this thesis, an inner product is a positive semi-definite, symmetric bilinear form, cf. Barrett et al. (2010a). Since tangential parts are not used in (3.6), we cannot conclude $\vec{\eta} = 0$ from $(\vec{\eta}, \vec{\eta})_{2,nor} = 0$.

A time-dependent function $\vec{x}$ is called solution of the gradient flow equation, if

$$(\vec{x}_t, \vec{\eta})_{2,nor} = -\left(\delta E(\Gamma)\right)(\vec{\eta}) \tag{3.7}$$

holds for all $\vec{\eta} : I \to \mathbb{R}^2$. This flow can be interpreted as a flow in the direction of the steepest descent.

For the classical active contours of Kass et al. (1988), the gradient flow equation (3.7) is dependent on the particular choice of the parameterization (cf. (3.5)), which is a disadvantage of this edge detection method. Two different parameterizations of the same initial curve can result in two different flows. Further, the fact that forth order derivatives are involved, poses a high regularity constraint on the curves.

The geometric active contours model of Malladi et al. (1995) and the geodesic active contours model of Caselles et al. (1997a) adopt the idea of the classical active contours of Kass et al. (1988) and perform detection of object boundaries (also called *shape recovery*, cf. Malladi et al. (1995)). In contrast to Kass et al. (1988), the corresponding flows are independent on the parameterization of the curve and depend only on geometrical quantities like curvature κ and normal vector field $\vec{\nu}$ of the curve.

Caselles et al. (1997a) pursue a variational approach and minimize an energy which is small where $\|\nabla u\|$ is big. The energy to be minimized is given by

$$E(\Gamma) = \int_\Gamma f \,\mathrm{d}s = \int_I f(\vec{x}) \|\vec{x}_\rho\| \,\mathrm{d}\rho, \tag{3.8}$$

where $f : \Omega \to \mathbb{R}$ is a smooth, positive edge indicator function, i.e. $f(\vec{z}) = g(\|\nabla u(\vec{z})\|)$ with $g : [0, \infty) \to \mathbb{R}$ strictly decreasing. Often, g is designed such that $g(0) = 1$. Again, for $\vec{\eta} : I \to \mathbb{R}^2$ we compute the first variation of $E(\Gamma)$ in the direction $\vec{\eta}$. Therefore, we choose $\vec{y} : I \times (-\epsilon_0, \epsilon_0) \to \mathbb{R}^2$ with $\vec{y}(\rho, 0) = \vec{x}(\rho)$ and $\vec{y}_\epsilon(\rho, 0) = \vec{\eta}(\rho)$ and we compute

$$\begin{aligned}
\left(\delta E(\Gamma)\right)(\vec{\eta}) &:= \left.\frac{\mathrm{d}}{\mathrm{d}\epsilon}\right|_{\epsilon=0} \int_I f(\vec{y}) \|\vec{y}_\rho\| \,\mathrm{d}\rho \\
&= \int_I \left(\nabla f(\vec{y}) \,.\, \vec{y}_\epsilon \,\|\vec{y}_\rho\| + f(\vec{y}) \frac{\vec{y}_\rho}{\|\vec{y}_\rho\|} \,.\, \vec{y}_{\rho\epsilon} \right)\Bigg|_{\epsilon=0} \mathrm{d}\rho \\
&= \int_I \left(\nabla f(\vec{x}) \,.\, \vec{\eta} \,\|\vec{x}_\rho\| + f(\vec{x}) \frac{\vec{x}_\rho}{\|\vec{x}_\rho\|} \,.\, \vec{\eta}_\rho \right) \mathrm{d}\rho \\
&= \int_\Gamma \left(\nabla f(\vec{x}) \,.\, \vec{\eta} + f(\vec{x}) \vec{x}_s \,.\, \vec{\eta}_s \right) \mathrm{d}s \\
&= \int_\Gamma \left(\nabla f(\vec{x}) - (\nabla f(\vec{x}) \,.\, \vec{x}_s) \vec{x}_s - f(\vec{x}) \vec{x}_{ss} \right) \,.\, \vec{\eta} \,\mathrm{d}s \\
&= \int_\Gamma \left(\vec{P}_n \,\nabla f(\vec{x}) - f(\vec{x}) \vec{x}_{ss} \right) \,.\, \vec{\eta} \,\mathrm{d}s.
\end{aligned} \tag{3.9}$$

We used integration by parts for the second last identity and the definition of the operator $\vec{P}_n$ for the last identity. The corresponding gradient flow equation is

$$(\vec{x}_t, \vec{\eta})_{2,nor} = -\int_\Gamma \left(\vec{P}_n \,\nabla f(\vec{x}) - f(\vec{x}) \vec{x}_{ss} \right) \,.\, \vec{\eta} \,\mathrm{d}s \quad \forall \vec{\eta} : I \to \mathbb{R}^2, \tag{3.10}$$

where $\vec{x} : I \times [0,T] \to \mathbb{R}^2$ is a time-dependent function. Let $\Gamma(t)$ be defined as the image of $\vec{x}(.,t)$ for $t \in [0,T]$. Let a unit normal field $\vec{\nu}(.,t)$ on $\Gamma(t)$ be defined by $\vec{\nu} := (\vec{x}_s)^\perp$, where $.^\perp$ denotes the counterclockwise rotation by $\frac{\pi}{2}$. We define the normal velocity of $\Gamma(t)$ by

$$V_n := \vec{x}_t \,.\, \vec{\nu}. \tag{3.11}$$

As we consider curves in $\mathbb{R}^2$, we have $V_n \vec{\nu} = \vec{P}_n \, \vec{x}_t$ and $\vec{P}_n \, \nabla f(\vec{x}) = (\nabla f(\vec{x}) \,.\, \vec{\nu}) \, \vec{\nu}$. From (3.10) and the identity

$$\kappa \, \vec{\nu} = \vec{x}_{ss} \tag{3.12}$$

(cf. (2.21)), we obtain

$$V_n = f\kappa - \nabla f \,.\, \vec{\nu}. \tag{3.13}$$

As claimed before, the evolution equation is independent on the particular parameterization: The edge indicator function f is evaluated at points on the curve and therefore only dependent on the image of $\vec{x}$ which is the current curve. Further only geometrical quantities like V_n, κ and $\vec{\nu}$ are involved.

The image segmentation works as follows: In homogeneous regions, where u is approximately constant, the gradient $\nabla u(\vec{x})$ is approximately zero. Consequently, $f(\vec{x}) \approx 1$ (using $g(0) = 1$) and $\nabla f(\vec{x}) \approx 0$. In these regions the flow reduces to the well-known curvature flow $V_n = \kappa$, also called curve shortening flow. The edge indicator function f is mainly *active* near edges which are classified by changes of the image function. At those edges $f \approx 0$ let the curve stop and the term $-\nabla f \,.\, \vec{\nu}$ pushes the curve to the middle of the edges, see also Caselles et al. (1997a), Figure 1.

If the initial contour is not near the edges, this method is very slow since the flow reduces to the curvature flow. To increase the speed a term of constant flow can be inserted to the evolution equation:

$$V_n = f \, (\kappa + c) - \nabla f \,.\, \vec{\nu}. \tag{3.14}$$

The constant $c \in \mathbb{R}$ is called *balloon* or *inflation* term (Cohen, 1991; Kichenassamy et al., 1996). It does not only increase the velocity of the flow, it does also allow a curve to expand, if $c < 0$ and if its absolute value is big compared to the curvature. One can show that a flow in normal direction can also be motivated by a variational approach. It is the gradient flow corresponding to a weighted area functional where the area of the open set enclosed by the curve is considered on the assumption that Γ is a closed curve without self-intersections. In detail, the evolution equation (3.14) can be derived by minimizing

$$E(\Gamma) = \int_\Gamma f \, \mathrm{d}s + \int_{\mathrm{Int}(\Gamma)} f \, c \, \mathrm{d}x, \tag{3.15}$$

where $\mathrm{Int}(\Gamma)$ denotes the interior of the curve, i.e. the subset of Ω enclosed by the curve. The functional (3.15) maps a curve to the sum of a weighted length functional and a weighted area functional. The evolution equation (3.14) can be derived similar as above for (3.13) by using methods from the theory of calculus of variations.

The geodesic active contours method is gradient-based which causes several problems additionally to the effort of the computation of the gradient ∇u: Noisy images can hardly be processed without a prior smoothing: At locations where the noise is high, the gradient of the image is high, and an evolving contour can stop at wrong, apparent edges which are only local minimums of the energy functional. To avoid false detection, the image intensity function u_0 needs to be smoothed. This causes a high additional computational effort. Moreover, it is not easy to determine an appropriate grade of smoothing: The denoising must be strong enough to smooth out noise. However, if the denoising is too strong, sharp edges will be smoothed

out and as a consequence the contour will not stop at region boundaries. This is also a drawback of the method in case of so-called weak edges, where the gradient of the intensity function changes only slightly. The moving contour will be decelerated at weak edges, but the contour may not stop and those edges may not be detected. Further, edge-based methods are not able to detect regions which consists of a grouping of smaller objects, cf. Aubert and Kornprobst (2006).

The constant c, the balloon term, must be chosen carefully. If it is zero or small, the evolution will be very slow. If it is too big, the curve will not stop at edges. Additionally, if the curvature term $f\,\kappa$ is dominated by the term $f\,c$, the smoothing effect caused by the curvature decreases. The choice of the sign of c further needs some pre-knowledge of the decomposition of the image. It is necessary to know if the initial contour lies inside or outside of the object to be detected. Its application on problems where the image segmentation should be performed autonomously without user-intervention is thus restricted. If part of the curve lies inside and part of the curve lies outside of the object, the geodesic active contours method with a constant balloon term will not work.

3.2.3 The Mumford-Shah Functional and Region-based Active Contours

Edge-based methods mainly depend on the image data along the contours. Only in case of a balloon term, information of the image inside of closed contours is used, cf. (3.15). Edge-based active contours aim at finding sharp edges with high gradient of the image function. Since partitions of an image are not necessarily defined by such sharp edges, region-based active contours models depend on image properties inside the image domain Ω. Region-based methods were studied by many authors, see for example Ronfard (1994); Chan and Vese (2001); Tsai et al. (2001).

Many region-based approaches use the image function u_0, not its gradient, to segment the image into homogeneous, connected regions. Often, it is the image function which characterizes connected regions, not its gradient ∇u_0, since the gradient characterizes only sharp edges. As a consequence, images with moderate noise can be segmented by region-based methods without any prior smoothing.

Before considering region-based approaches which segment the image in partitions or regions, we first revisit the Mumford-Shah model, see Chapter 1. Tsai et al. (2001) and Chan and Vese (2001) apply the model of Mumford and Shah (1989) for image segmentation and for image denoising by a piecewise smooth or constant function.

The original Mumford-Shah method for optimal approximation of images aims at finding a set of curves $\Gamma = \Gamma_1 \cup \ldots \cup \Gamma_{N_C}$ and a piecewise smooth function $u : \Omega \to \mathbb{R}$ with possible discontinuities across Γ approximating the original image u_0. The energy to be minimized is

$$E^{\mathrm{MS}}(u, \Gamma) = \sigma|\Gamma| + \int_{\Omega\setminus\Gamma} \|\nabla u\|^2 \,\mathrm{d}x + \lambda \int_{\Omega} (u_0 - u)^2 \,\mathrm{d}x, \tag{3.16}$$

where $\sigma, \lambda > 0$ are weighting parameters and $|\Gamma|$ denotes the total length of the curves in Γ, see Section 2.1.1. (For non-smooth curves, we identify $|\Gamma|$ with the one-dimensional Hausdorff measure of $\Gamma \subset \mathbb{R}^2$.)

The first term in (3.16) penalizes the length of the curves, the second term does not allow u to change much in $\Omega \setminus \Gamma$, and the third term requests that u is a good approximation of u_0.

Remark 3.1. *(i) The general Mumford-Shah problem is difficult to solve. The main difficulty of the problem is the fact that it involves an unknown function u defined on a two-dimensional set and an unknown one-dimensional set Γ, cf. Aubert and Kornprobst (2006); Doğan et al. (2008).*

(ii) Mumford and Shah (1989) conjectured that a minimizer of E^{MS} exists such that Γ consists of a finite set of $C^{1,1}$-arcs. The curves are either closed or have endpoints which belong to triple junctions or which are free endpoints, so-called crack-tips*, or which meet perpendicularly the image boundary $\partial\Omega$.*

(iii) The Mumford-Shah functional can be used to segment a given image in disjoint regions separated by interfaces. However, in the Mumford-Shah setup, the curves can also be open contours with free endpoints. Therefore, the Mumford-Shah functional can be used for both finding partitions of similar image intensity (regions) and finding discontinuities of the image function (edges).

(iv) The Mumford-Shah conjecture is difficult to prove, see Aubert and Kornprobst (2006). We first consider a possible solution (u, Γ), $\Gamma \subset \Omega$ closed in $\Omega \subset \mathbb{R}^2$, with $|\Gamma| < \infty$, and $u \in W^{1,2}(\Omega \setminus \Gamma) \cap L^\infty(\Omega)$. One may ask whether existence of a minimizer (u, Γ) of the Mumford-Shah functional can be shown using the direct method in the calculus of variations. For that, we need to prove compactness and lower semicontinuity for a minimizing sequence. One can show that for a Borel set $\Omega_0 \subset \mathbb{R}^2$ with topological boundary $\partial\Omega_0$, the mapping $\Omega_0 \mapsto |\partial\Omega_0|$ is not *lower semicontinuous with respect to any compact topology, see Aubert and Kornprobst (2006). However, one can consider a reformulation of the Mumford-Shah problem and can consider functions in $SBV(\Omega)$, i.e. in the space of special functions of bounded variation. For a function $u \in SBV(\Omega)$, let S_u denote the jump set of u. Instead of* (3.16)*, one can prove the existence of a minimizer of*

$$G(u) = \sigma|S_u| + \int_\Omega \|\nabla u\|^2 \,\mathrm{d}x + \lambda \int_\Omega (u_0 - u)^2 \,\mathrm{d}x, \quad u \in SBV(\Omega) \cap L^\infty(\Omega), \tag{3.17}$$

by showing compactness and lower semicontinuity, see also Ambrosio (1990). Having found a solution $u \in SBV(\Omega) \cap L^\infty(\Omega)$ of (3.17)*, we can set $\Gamma = \Omega \cap \overline{S_u}$.*

Now we want to approximate solutions of the Mumford-Shah problem. Similarly to Chan and Vese (2001), we consider a reduced problem for the segmentation of the image: We assume that the curves in Γ partition the set Ω in connected segments. In Section 3.2.10, we propose a method how the case including curves with free endpoints can be handled. For the moment, we consider curves which are either closed or which endpoints belong to a triple junction or to the boundary of Ω.

Let $\Omega_1, \ldots, \Omega_{N_R}$ be the components of $\Omega \setminus \Gamma$. Further, we let each curve Γ_i be an interface-curve between two regions $\Omega_{k^+(i)}$ and $\Omega_{k^-(i)}$ with $k^+(i), k^-(i) \in \{1, \ldots, N_R\}$. Thus, we have the following decomposition of the domain Ω:

$$\Omega = \Omega_1 \cup \ldots \cup \Omega_{N_R} \cup \Gamma_1 \cup \ldots \cup \Gamma_{N_C}. \tag{3.18}$$

Figure 3.2 gives an example of a decomposition of a rectangular domain. The regions and interfaces can have more than one connected component. The interfaces can be closed, they can meet at a triple junction, and boundary intersections with the image boundary $\partial\Omega$ can occur.

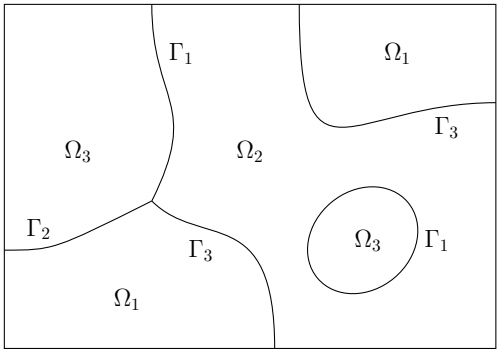


Figure 3.2: Example of a decomposition in regions Ω_k and interfaces Γ_i, $k = 1, \dots N_R = 3$, $i = 1, \dots, N_C = 3$.

First, we restrict ourselves to piecewise constant functions u of the form

$$u = \sum_{k=1}^{N_R} c_k \chi_{\Omega_k}, \tag{3.19}$$

where χ_{Ω_k} denotes the characteristic function of Ω_k. As u is piecewise constant, we have $\nabla u = 0$ on $\Omega \setminus \Gamma$, and the functional (3.16) reduces to

$$E(\Gamma, c_1, \dots, c_{N_R}) = \sigma|\Gamma| + \lambda \sum_{k=1}^{N_R} \int_{\Omega_k} (u_0 - c_k)^2 \,\mathrm{d}x. \tag{3.20}$$

The length term $\sigma|\Gamma|$ ensures that the curves $\Gamma_1, \dots, \Gamma_{N_C}$ have finite length. It is also called *perimeter* energy. The second term is a *bulk* energy, designed such that the sets Ω_k, $k = 1, \dots, N_R$, best approximate the several objects in the image. The interfaces Γ_i, $i = 1, \dots, N_C$, are attracted to the edges of objects in the image.

We first consider a setup of two disjoint phases and one interfacing, closed curve, i.e. $N_R = 2$ and $N_C = 1$. We will derive the evolution law as gradient flow. In Section 3.2.5 and 3.2.6 we will state how to segment an image consisting of multiple regions ($N_R \geq 2$ arbitrary) and how to handle open curves which endpoints meet at triple junctions or meet the image boundary $\partial\Omega$.

Let $\Gamma \subset \Omega$ be a closed curve without self-intersections and let $\vec{x} : I \to \mathbb{R}^2$, $I = S^1 \cong \mathbb{R}/\mathbb{Z}$, be a smooth parameterization of Γ. The curve divides Ω in two disjoint sets $\Omega_1, \Omega_2 \subset \Omega$ such that

$$\Omega = \Omega_1 \cup \Omega_2 \cup \Gamma, \quad \partial\Omega_1 = \Gamma, \quad \Omega_2 = \Omega \setminus \overline{\Omega_1}.$$

Chan and Vese (2001) consider an energy similar to (3.20) which is of the form

$$E(\Gamma, c_1, c_2) = \sigma \int_I \|\vec{x}_\rho\| \,\mathrm{d}\rho + \mu \int_{\Omega_1} 1 \,\mathrm{d}x + \lambda_1 \int_{\Omega_1} f_1 \,\mathrm{d}x + \lambda_2 \int_{\Omega_2} f_2 \,\mathrm{d}x, \tag{3.21}$$

where $\sigma, \lambda_1, \lambda_2 > 0$, $\mu \geq 0$ are weighting parameters and $f_k : \Omega_k \to \mathbb{R}$, $k = 1, 2$, are defined by

$$f_k(\vec{z}) := (u_0(\vec{z}) - c_k)^2. \tag{3.22}$$

For $\mu = 0$ and $\lambda_1 = \lambda_2 = \lambda$, this is the functional (3.20) with $N_R = 2$.

The first two integrals in (3.21) are the length of the curve and the area enclosed by the curve. The last two integrals can be interpreted as external energies.

It has to be noted, that f_1 and f_2 in (3.22) do not depend on the gradient in contrast to edge-based methods, see Section 3.2.2. Only the raw image intensity function u_0 is used. This is why no prior smoothing of u_0 needs to be done.

We now apply methods from the calculus of variations to obtain an evolution equation for Γ and equations for the constants c_1 and c_2. We do not search for a minimizer (Γ, c_1, c_2) of the functional (3.21) in one step. We perform a two-step approach, see also Chan and Vese (2001): Starting with an initial curve, we fix the curve and consider variations in c_1 and c_2 and derive necessary conditions for the coefficients such that the energy is minimized. After having computed the coefficients c_k, $k = 1, 2$, we fix these quantities and consider a variation in Γ and derive an evolution equation. An update of the curve Γ, changes the regions Ω_1 and Ω_2. In the next step, c_1 and c_2 are recomputed using the new regions, followed by an update of Γ using the new values of c_k, $k = 1, 2$. This is done iteratively until the velocity of the curve is almost zero, i.e. until the curve has approached a stationary solution.

In the following, we present details of the calculation. First, we fix Γ and consider a variation in $c_k \in \mathbb{R}$, $k \in \{1, 2\}$. This leads to

$$c_k = \frac{\int_{\Omega_k} u_0 \,\mathrm{d}x}{\int_{\Omega_k} 1 \,\mathrm{d}x}, \tag{3.23}$$

i.e. $c_k \in \mathbb{R}$ is set to the mean of u_0 in Ω_k.

In contrast to Chan and Vese (2001) who use the level set method of Osher and Sethian (1988) and derive the evolution equations for the corresponding level set function, we use a parametric description of the curve. We refer to Barrett et al. (2010a) who derive geometric evolution equations for certain flows, i.a. for curve shortening flow.

We fix c_k, $k = 1, 2$. For $\vec{\eta} : I \to \mathbb{R}^2$, we choose $\vec{y} : I \times (-\epsilon_0, \epsilon_0) \to \mathbb{R}^2$ with $\vec{y}(\rho, 0) = \vec{x}(\rho)$, $\vec{y}_\epsilon(\rho, 0) = \vec{\eta}(\rho)$, cf. Barrett et al. (2010a). Let Γ^ϵ denotes the image of $\vec{y}(., \epsilon)$, and Ω_1^ϵ, Ω_2^ϵ the interior and exterior of Γ^ϵ, respectively. Further, $\vec{\nu}^\epsilon : I \to \mathbb{R}^2$ denotes an inner normal vector field at Γ^ϵ, i.e. $\vec{\nu}^\epsilon(\rho)$ is an inner normal vector of Γ^ϵ at the point $\vec{y}(\rho, \epsilon)$. For fixed ρ the velocity of $\epsilon \mapsto \vec{y}(\rho, \epsilon)$ in normal direction $\vec{\nu}^\epsilon(\rho)$ is given by $\vec{y}_\epsilon(\rho, \epsilon) \,.\, \vec{\nu}^\epsilon(\rho)$. We have $\Gamma^0 = \Gamma$, $\Omega_i^0 = \Omega_i$ and $\vec{\nu}^0 =: \vec{\nu}$.

The first variation of $\vec{x} \mapsto E(\vec{x}(I), c_1, c_2)$ in the direction $\vec{\eta}$ is

$$\begin{aligned}
(\delta E(\Gamma))(\vec{\eta}) &= \frac{\mathrm{d}}{\mathrm{d}\epsilon}\bigg|_{\epsilon=0} \left(\sigma \int_I \|\vec{y}_\rho\| \,\mathrm{d}\rho + \mu \int_{\Omega_1^\epsilon} 1 \,\mathrm{d}x + \lambda_1 \int_{\Omega_1^\epsilon} f_1 \,\mathrm{d}x + \lambda_2 \int_{\Omega_2^\epsilon} f_2 \,\mathrm{d}x \right) \\
&= \left(\sigma \int_I \frac{\vec{y}_\rho}{\|\vec{y}_\rho\|} \,.\, \vec{y}_{\rho\epsilon} \,\mathrm{d}\rho + \mu \int_I (-\vec{y}_\epsilon \,.\, \vec{\nu}^\epsilon) \|\vec{y}_\rho\| \,\mathrm{d}\rho + \lambda_1 \int_I f_1(\vec{y})(-\vec{y}_\epsilon \,.\, \vec{\nu}^\epsilon) \|\vec{y}_\rho\| \,\mathrm{d}\rho + \right. \\
&\quad \left. + \lambda_2 \int_I f_2(\vec{y})(\vec{y}_\epsilon \,.\, \vec{\nu}^\epsilon) \|\vec{y}_\rho\| \,\mathrm{d}\rho \right)\bigg|_{\epsilon=0} \\
&= \sigma \int_\Gamma \vec{x}_s \,.\, \vec{\eta}_s \,\mathrm{d}s + \int_\Gamma (-\mu - \lambda_1 f_1 + \lambda_2 f_2) \,\vec{\nu} \,.\, \vec{\eta} \,\mathrm{d}s \\
&= \int_\Gamma \left(-\sigma \vec{x}_{ss} + (-\mu - \lambda_1 f_1 + \lambda_2 f_2) \,\vec{\nu} \right) \,.\, \vec{\eta} \,\mathrm{d}s.
\end{aligned} \tag{3.24}$$

In the second line a transport theorem is applied (cf. Theorem 2.25) and in the last line integration by parts is performed (cf. Theorem 2.22). The corresponding gradient flow for a

time-dependent $\vec{x}$ is

$$(\vec{x}_t, \vec{\eta})_{2,nor} = -\int_\Gamma \left(-\sigma\vec{x}_{ss} + \left(-\mu - \lambda_1 f_1 + \lambda_2 f_2\right)\vec{\nu}\right) . \vec{\eta}\, \mathrm{d}s \quad \forall \vec{\eta} : I \to \mathbb{R}^2. \tag{3.25}$$

Using (3.25), the identity (3.12) and $\vec{P}_n^T = \vec{P}_n$, $\vec{P}_n^2 = \vec{P}_n$ we get

$$\vec{P}_n \vec{x}_t = (\sigma\kappa + \mu + \lambda_1 f_1 - \lambda_2 f_2)\vec{\nu}. \tag{3.26}$$

Defining the normal velocity by $V_n = \vec{x}_t \,.\, \vec{\nu}$ (cf. Definition 2.24) and using $\vec{P}_n\vec{x}_t = V_n\vec{\nu}$, we obtain the following law for the normal velocity, cf. Chan and Vese (2001):

$$V_n = \sigma\kappa + \mu + \lambda_1 f_1 - \lambda_2 f_2. \tag{3.27}$$

Inserting (3.22), the definition of f_i, we get

$$V_n = \sigma\kappa + \mu + \lambda_1 (u_0 - c_1)^2 - \lambda_2 (u_0 - c_2)^2. \tag{3.28}$$

Note, that the coefficients c_k, $k = 1, 2$, are now time-dependent. They are the mean of u_0 in $\Omega_k(t)$, $t \in [0, T]$.

In case $\mu = 0$ and $\lambda_1 = \lambda_2 = \lambda$, the evolution equation reduces to

$$V_n = \sigma\kappa + F, \tag{3.29a}$$

$$F = \lambda\left((u_0 - c_1)^2 - (u_0 - c_2)^2\right), \tag{3.29b}$$

which is the evolution equation corresponding to (3.20) for $N_R = 2$. Without the external energy term, the evolution would reduce to the well-known curvature flow (weighted with some constant $\sigma > 0$).

In summary, we restate the main advantages of region-based methods, like the Chan-Vese method, compared to edge-based methods:

- the gradient of the image intensity function need not be computed,
- prior smoothing of the raw image intensity function is not necessary,
- noisy images can be handled since the gradient is not involved,
- images with weak edges can be segmented.

For the last item, we recall that region-based methods aim at finding a composition of Ω into regions and an approximation u of u_0 such that u best approximates u_0 in the regions. Therefore, the segmentation depends on the values of u_0 inside the regions, not on the gradient of u_0 at edges, i.e. at the boundaries of the regions. Thus, there can also be smooth transitions (*weak edges*) between two gray values at the boundaries.

Because of the mentioned advantages, we focus on region-based methods in the following sections.

In addition, we note that a curve evolution is given by its velocity in normal direction V_n. Analytically, the same evolution is given by the scheme

$$\vec{x}_t = \sigma\vec{x}_{ss} + F\vec{\nu}. \tag{3.30}$$

However, this scheme restricts the velocity vector to point in normal direction, i.e. the tangential velocity is zero. In our scheme (3.29), the tangential velocity is a free parameter.

We will later discuss an advantage of our scheme: Using a spatial discretization, where the curve is approximated by a polygonal curve given by a finite set of so-called nodes or mesh points, our scheme provides equidistribution of the mesh points along the curve. This is presented in detail in Section 3.3.4, where we consider a corresponding semidiscrete scheme which is continuous in time and discrete in space. Our scheme leads to a tangential velocity such that the nodes are automatically redistributed in tangential direction leading to equal distances between the mesh points.

3.2.4 Representation of Curves and Related Works

In Section 3.2.2 and 3.2.3, we derived evolution equations as gradient flows of certain energy functionals. The evolution equation (3.13) associated with the geodesic active contours method and the evolution equations (3.28) and (3.29) associated with Chan-Vese method are independent on the special choice of the parameterization of the involved curves.

As stated in Section 2.1.2, there are several possibilities to represent a curve. An appropriate method to describe the evolution of curves has to satisfy the following requirements:

- support of multiple regions / phases,
- support of complex networks of curves with triple junctions, intersections with the image boundary and free endpoints,
- good mesh quality when using a spatial discretization for a numerical approximation,
- handling of topology changes like splitting of a curve, merging of curves, creation of triple junctions and boundary intersection points.

For image processing purposes and in particular for active contours methods for image segmentation, the level set method developed by Osher and Sethian (1988) is applied by many authors, e.g. Malladi et al. (1995), Caselles et al. (1997a), Kichenassamy et al. (1996), Chan and Vese (2001), Tsai et al. (2001), Sapiro (2006) to mention a few. In level-set methods, a hypersurface is embedded as the zero level set of a function defined on the image domain Ω, cf. Section 2.1.2. Corresponding equations for the level set function are derived from the evolution equations.

Ambrosio and Tortorelli (1990) propose approximations of the Mumford-Shah functional by elliptic functionals which contain a certain phase field energy. They use a phase field model to describe the edge set.

A phase field approach is also used by Beneš et al. (2004). The geodesic active contours model is reformulated to an equation for the unknown phase field function. This results in a modified Allen-Cahn equation. The problem is discretized and solved with finite differences.

For convex minimization problems we mention Chan et al. (2006) which try to find *global* minimizers of segmentation models and use level sets for the numerical solution. Bresson et al. (2007) computes global minimizers of active contours models. The authors use a convex regularization of the active contours models and use a dual formulation of the problem for the numerical solution. Moreover, convex relaxation approaches for computing approximate global solutions for minimal partitions problems (cf. Pock et al. (2009a); Chambolle et al. (2012)) and approximate global minimizers of the Mumford-Shah functional (cf. Pock et al. (2009b)) have been proposed. These approaches are based on functional lifting, i.e.

binary functions defined on a higher dimensional space are used for the multiphase labeling/segmentation. This method leads to convex minimization problems which are solved with a primal-dual algorithm. The boundaries of the regions are indirectly handled by the discontinuity set of a function $u \in SBV(\Omega)$. Also triple junctions (cf. Chambolle et al. (2012)) and free endpoints/cracks (cf. Pock et al. (2009b)) can be handled with their approach.

Also parametric methods are used in the literature. Mikula and Urbán (2012) make use of a parametric description of the curve and solve a system of partial differential equations with finite volume methods, see also Mikula and Ševčovič (2004a,b, 2006). Doğan et al. (2008) use a parametric method for image segmentation with the Mumford-Shah functional. They perform a variational shape optimization approach which leads to non-linear flows. A coupled system for the curvature κ, curvature vector $\vec{\kappa} = \kappa\vec{\nu}$, normal velocity V_n and velocity vector $\vec{V}_n = V_n\vec{\nu}$ is solved, see also Bänsch et al. (2005). A parametric approach is also used in Beneš et al. (2008) who propose a region-based active contours method of the form $V_n = \kappa + F$ with an alternative forcing term F compared to (3.29b).

In this thesis, a parametric approach based on Barrett et al. (2007a,b) is pursued. The evolution equations can be rewritten to a scheme for the parameterization $\vec{x}$ and the curvature κ by using the relation $V_n = \vec{x}_t \,.\, \vec{\nu}$ and the identity $\vec{x}_{ss} = \kappa\vec{\nu}$ as in Dziuk (1991). In Section 3.2.5 and 3.2.6, we state how multiple phases and complex curve networks can be handled. Multiple curves which separate multiple phases are parameterized. Endpoints of non-closed curves can meet at triple junctions or can meet the outer boundary $\partial\Omega$. Additional conditions have to be satisfied at triple junctions and boundary intersection points. The method can be extended to cover also curves with free endpoints, see Section 3.2.10.

In principle, it is possible to apply level set techniques for problems including triple junctions. Vese and Chan (2002) make use of two level set functions to handle a triple junction. The idea of using two such functions Φ and Ψ is illustrated in Figure 3.3. An equation for each level set function has to be solved. The phases are finally identified by considering intersections like $\{\Phi < 0\} \cap \{\Psi < 0\} \subset \Omega$. Triple junctions are not explicitly represented. At one edge in Figure 3.3, the zero level sets of Φ and Ψ overlap. The corresponding curve is not uniquely represented by one single zero level set which is a drawback of the method. With n level set functions, 2^n regions can be handled using the method of Vese and Chan (2002). The computational effort primarily depends on the number of different regions, not on the length of the curves as in the parametric case. Therefore multiphase segmentation results in an increased computational effort compared to two-phase segmentation. More recently, a new approach has been proposed by Dubrovina et al. (2013), where multiphase image segmentation is performed by using a single, non-negative level set function. The approach, based on the Voronoi Implicit Interface Method (cf. Saye and Sethian (2011)), makes use of so-called ϵ-level sets and reconstruction of the curve followed by an update of the level set function. Also junctions can be handled with this new multi-region approach of Saye and Sethian (2011) for level sets.

Good mesh quality is an important issue when using a parametric approach. Numerically, smooth curves are approximated by polygonal curves, see Section 3.3. The node points of the polygonal curves should be equally distributed along the curve. However, the direct parametric approach often bunches some mesh points together whereas other points drift apart, see e.g. Srikrishnan et al. (2007). Consequently, quantities like the curvature are often not computed accurately and also the segmentation of the regions can be too rough if the local node distance is too large. This problem has been addressed in the context of the Mumford-Shah formulation by Cremers et al. (2001) who modified the usual length constrained on the contour by using a so-called diffusion snake. Other authors (Mikula and Ševčovič, 2004a; Srikrishnan et al., 2007; Doğan et al., 2008; Beneš et al., 2008; Ševčovič and Yazaki, 2011;

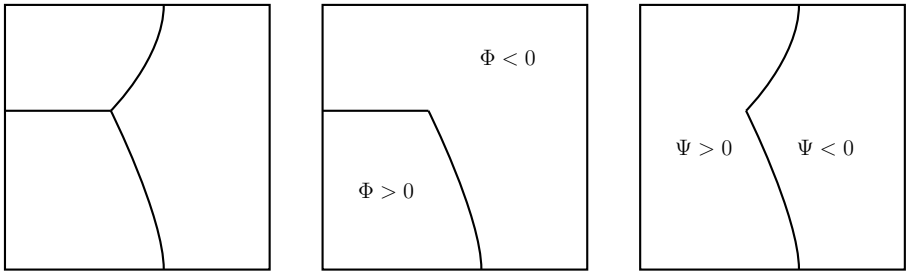


Figure 3.3: Example how a triple junction can be handled by using two level set functions Φ and Ψ, cf. Vese and Chan (2002).

Mikula and Urbán, 2012) use special routines or design tangential flows to prevent numerical instabilities and to achieve equidistribution of nodes and thus a good mesh quality. Often a curvature adjusted tangential velocity is used, see Mikula and Ševčovič (2004a). In this thesis, we make use of the scheme of Barrett et al. (2007a,b) which has good properties with respect to equidistribution of mesh points. This property is built into their scheme such that no special routine for redistribution of mesh points is necessary. The equidistribution property is analyzed and discussed in detail in Section 3.3.4.

Standard parametric approaches cannot handle topology changes automatically in contrast to level set methods. This fact seems to be the main drawback of parametric methods. The ability to detect topology changes is important for image segmentation applications. Without any pre-knowledge or user-input concerning the decomposition of the image, one can start the segmentation technique with one or more closed curves which evolve in time. For instance, a new interface and two new triple junctions occur when two different curves meet. This has to be detected and a change in topology has to be performed. The necessary modifications can include changing the neighbor relations between mesh points, creating new nodes and entire curves, deleting nodes and curves, transition from a closed curve to an open curve or vice versa. Still, the main computational effort is the *detection* of a topology change. For that, there exist different suggestions in the literature, see Araki et al. (1997); Doğan et al. (2008); Nakhmani and Tannenbaum (2012). In Section 3.3.5, we present a method for detection of topology changes which is based on the method of Balažovjech et al. (2012) and Mikula and Urbán (2012) with some extensions to multiple phases and curves with triple junctions and boundary intersection points. It results in an efficient detection of topology changes with an effort of $\mathcal{O}(N)$, where N is the number of mesh points in the curve network.

An important advantage of parametric methods is, that only one-dimensional geometric partial differential equations have to be solved in contrast to standard level set or phase field methods. The corresponding discretized versions of the one-dimensional equations can be solved very fast with a direct numerical algebra solver like the UMFPACK algorithm, see Davis (2004). Level set and phase field methods result in problems of higher dimension. This leads to a higher computational effort or requires special techniques like the narrow band algorithm, see Adalsteinsson and Sethian (1995); Sethian (1999).

Barrett et al. (2014) compared parametric sharp interface methods with phase field methods by performing quantitative comparisons and numerical experiments. They considered in particular the Mullins-Sekerka and the isotropic and anisotropic Stefan problem including dendritic growth of hypersurfaces. One main conclusion of their work is that parametric methods are more accurate and at the same time more efficient from a computational point of view compared to phase field methods. This motivates to pursue a parametric approach

also for image segmentation problems.

In summary, the evolution equations can be solved by using various techniques. We decided to pursue a parametric approach because of several advantages. First, it is a direct, one-dimensional and thus fast method. Second, the parametric method can be extended to multiple phases and complex curve networks including triple junctions, intersections with the outer boundary and free endpoints/crack tips. The necessary framework will be developed in the next sections. Topology changes can be efficiently detected by an extension of the method of Balažovjech et al. (2012) and Mikula and Urbán (2012), see Section 3.3.5.

3.2.5 Multiphase Image Segmentation

The image segmentation model for two phases (3.29) can be generalized to multiple phases separated by multiple open or closed curves. For that we consider a decomposition of the image domain into N_R regions $\Omega_1, \dots, \Omega_{N_R}$ separated by N_C curves $\Gamma_1, \dots, \Gamma_{N_C}$, recall (3.18).

Fixing the union of curves $\Gamma = \Gamma_1 \cup \dots \cup \Gamma_{N_C}$ in (3.20) and considering a variation in c_k, $k \in \{1, \dots, N_R\}$, leads to

$$c_k = \frac{\int_{\Omega_k} u_0 \,\mathrm{d}x}{\int_{\Omega_k} 1 \,\mathrm{d}x}, \tag{3.31}$$

i.e. $c_k \in \mathbb{R}$ is set to the mean of u_0 in Ω_k.

For a variation of the contours belonging to Γ, we consider time-dependent curves and regions: Let $\Gamma_1(t), \dots, \Gamma_{N_C}(t)$, $t \in [0,T]$, be smooth, evolving curves. Let an approximation $u(t)$ of u_0 be given by a piecewise constant function with $u(t)_{|\Omega_k(t)} = c_k(t) \in \mathbb{R}$, $k = 1, \dots, N_R$, where $c_k(t)$ is set to the mean of u_0 in $\Omega_k(t)$. The curve $\Gamma_i(t)$ should evolve in time such that the energy (3.20) decreases most quickly.

We now introduce a representation of the curves $\Gamma_i(t)$, $i = 1, \dots, N_C$, by smooth parameterizations. Let $\vec{x}_i : I_i \times [0,T] \to \mathbb{R}^2$, $t \in [0,T]$, be a smooth mapping such that $\vec{x}_i(\,.\,,t)$ is a smooth parameterization of $\Gamma_i(t)$. The set I_i is a one-dimensional reference manifold, e.g. $I_i = [0,1]$ for open curves, i.e. curves with $\partial\Gamma_i(t) \neq \emptyset$, and $I_i = S^1 \cong \mathbb{R}/\mathbb{Z}$ for closed curves, i.e. curves with $\partial\Gamma_i(t) = \emptyset$. Further, we define a normal vector field $\vec{\nu}_i(\,.\,,t)$ on $\Gamma_i(t)$ by $\vec{\nu}_i : I_i \times [0,T] \to \mathbb{R}^2$ such that $\vec{\nu}_i(\rho,t)$ is a normal on $\Gamma_i(t)$ at $\vec{x}_i(\rho,t)$. To be precise, we set

$$\vec{\nu}_i(\rho,t) = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} (\vec{x}_i)_s(\rho,t), \tag{3.32}$$

where s is the arc-length and $(\vec{x}_i)_s = (\vec{x}_i)_\rho / \|(\vec{x}_i)_\rho\|$ denotes the derivative of $\vec{x}_i$ with respect to arc-length.

Let $k^+(i), k^-(i) \in \{1, \dots, N_R\}$ denote the indices of the two regions which are separated by $\Gamma_i(t)$. We choose the parameterization such that the normal vector field defines an orientation of $\Gamma_i(t)$ with $\vec{\nu}_i(\,.\,,t)$ pointing from phase $\Omega_{k^-(i)}(t)$ to $\Omega_{k^+(i)}(t)$.

Similar to (3.29), using methods from the calculus of variations, we obtain the following evolution law for curves $\Gamma_i(t)$:

$$(V_n)_i = \sigma\kappa_i + F_i, \quad i = 1, \dots, N_C, \tag{3.33}$$

where $(V_n)_i = (\vec{x}_i)_t \,.\, \vec{\nu}_i$ denotes the normal velocity and κ_i the curvature of $\Gamma_i(t)$ and F_i is an external forcing term defined by

$$F_i(\,.\,,t) = \lambda[(u_0 - c_{k^+(i)}(t))^2 - (u_0 - c_{k^-(i)}(t))^2]. \tag{3.34}$$

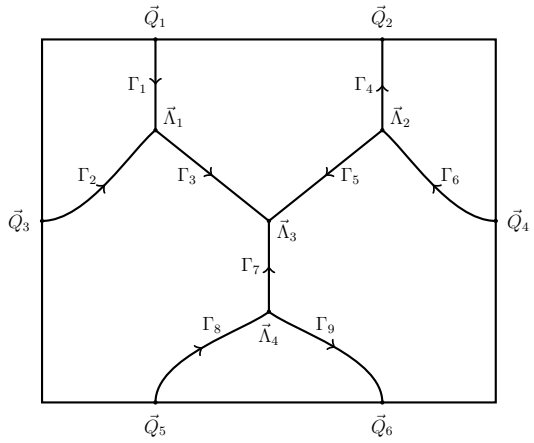


Figure 3.4: Example of a curve network with triple junctions and boundary intersection points.

Using the parametric description, the evolution equation (3.33) can be rewritten as

$$(\vec{x}_i)_t \,.\, \vec{\nu}_i = \sigma\kappa_i + F_i, \tag{3.35a}$$

and the curvature $\kappa_i : I_i \times [0,T] \to \mathbb{R}$ is related to $\vec{x}_i$ by

$$\kappa_i \vec{\nu}_i = (\vec{x}_i)_{ss}, \quad i = 1, \ldots, N_C. \tag{3.35b}$$

3.2.6 Triple Junctions and Intersections with the Image Boundary

The methods presented above can be generalized to complex structures of curves which involve triple junctions and boundary intersection points. At these points, additional conditions need to be stated, cf. Barrett et al. (2007a). Figure 3.4 shows an example of a curve network with triple junctions and boundary intersection points.

Let Γ_i be a curve with a smooth parameterization $\vec{x}_i : I_i \to \mathbb{R}^2$, $i = 1, \ldots, N_C$. Let $\vec{\Lambda}_k \in \Omega$, $k = 1, \ldots, N_T$, denote the triple junctions. For each $k \in \{1, \ldots, N_T\}$, let $i_{k,1}, i_{k,2}, i_{k,3} \in \{1, \ldots, N_C\}$ denote the indices of nonclosed curves $\Gamma_{i_{k,l}}$ with $I_{i_{k,l}} = [0,1]$, $l = 1,2,3$, $i_{k,1} \neq i_{k,2} \neq i_{k,3} \neq i_{k,1}$, such that

$$\vec{x}_{i_{k,1}}(\rho_{k,1}) = \vec{x}_{i_{k,2}}(\rho_{k,2}) = \vec{x}_{i_{k,3}}(\rho_{k,3}) = \vec{\Lambda}_k,$$

where $\rho_{k,l} \in \{0,1\}$ corresponds to the start or end point of the curve $\Gamma_{i_{k,l}}$, $l = 1,2,3$.

At the triple junctions $\vec{\Lambda}_k$, $k = 1, \ldots, N_T$, an attachment condition and Young's law need to hold:

$$\text{the triple junction } \vec{\Lambda}_k \text{ does not pull apart,} \tag{3.36a}$$

$$\sum_{l=1}^{3} (-1)^{\rho_{k,l}} \, \vec{\tau}_{i_{k,l}}(\rho_{k,l}) = 0, \tag{3.36b}$$

where $\vec{\tau}_{i_{k,l}} := (\vec{x}_{i_{k,l}})_s$ is a tangent vector field at $\Gamma_{i_{k,l}}$, $l = 1,2,3$. The condition (3.36b) is equivalent to a 120° angle condition at triple junctions, see Barrett et al. (2007a).

Let $\vec{Q}_k \in \partial\Omega$, $k = 1, \dots, N_I$, be the set of boundary intersection points. For $k = 1, \dots, N_I$, let $i_{I,k}$ denote the curve index and $\rho_{I,k} \in \{0, 1\}$ such that $\vec{x}_{i_{I,k}}(\rho_{I,k}) = \vec{Q}_k$. Let $\vec{\tau}_{i_{I,k}} := (\vec{x}_{i_{I,k}})_s$ be the corresponding tangent vector field at $\Gamma_{i_{I,k}}$. The following conditions need to hold at the boundary intersection points $\vec{Q}_k$, $k = 1, \dots, N_I$:

$$\text{the curve endpoint } \vec{Q}_k \text{ remains attached to } \partial\Omega, \tag{3.37a}$$

$$\vec{\tau}_{i_{I,k}}(\rho_{I,k}) \,.\, \vec{n}_{\partial\Omega}(\vec{Q}_k)^{\perp} = 0, \tag{3.37b}$$

where $\vec{n}_{\partial\Omega}$ is a normal vector field at the boundary of the rectangular image domain Ω. The first equation is an attachment condition, and the second equation enforces a $90°$ angle condition at the boundary intersection point.

3.2.7 Weak Scheme

We introduce a weak scheme for (3.35) with (3.36) and (3.37) in case of triple junctions and boundary intersections.

As above, I_i denotes a one-dimensional reference manifold, e.g. $I_i = [0, 1]$ for open curves and $I_i = \mathbb{R}/\mathbb{Z}$ for closed curves. Let $\vec{x}_i : I_i \to \mathbb{R}^2$ be a parameterization of a curve Γ_i, $i = 1, \dots, N_C$. For triple junctions and boundary intersection points, the notation of Section 3.2.6 is adopted.

We define the following function spaces

$$W := H^1(I_1, \mathbb{R}) \times \dots \times H^1(I_{N_C}, \mathbb{R}), \tag{3.38a}$$

$$\underline{V} := \big\{(\vec{\eta}_1, \dots, \vec{\eta}_{N_C}) \in H^1(I_1, \mathbb{R}^2) \times \dots \times H^1(I_{N_C}, \mathbb{R}^2) \,:\, \vec{\eta}_{i_{k,1}}(\rho_{k,1}) = \\ = \vec{\eta}_{i_{k,2}}(\rho_{k,2}) = \vec{\eta}_{i_{k,3}}(\rho_{k,3}),\ \forall k = 1, \dots, N_T\big\}, \tag{3.38b}$$

$$\underline{V}_\partial := \big\{(\vec{\eta}_1, \dots, \vec{\eta}_{N_C}) \in \underline{V} \,:\, \vec{\eta}_{i_{I,k}}(\rho_{I,k}) \,.\, \vec{n}_{\partial\Omega}(\vec{x}_{i_{I,k}}(\rho_{I,k})) = 0,\ \forall k = 1, \dots, N_I\big\}. \tag{3.38c}$$

For scalar and vector-valued functions $u = (u_1, \dots, u_{N_C}), v = (v_1, \dots, v_{N_C}) \in L^2(I_1, \mathbb{R}^{(2)}) \times \dots \times L^2(I_{N_C}, \mathbb{R}^{(2)})$, we define

$$\int_\Gamma u \,.\, v \,\mathrm{d}s := \sum_{i=1}^{N_C} \int_{I_i} u_i \,.\, v_i \,\|(\vec{x}_i)_\rho\| \,\mathrm{d}\rho.$$

Let $\vec{\nu}_i$ be a normal vector field on Γ_i given by $\vec{\nu}_i = (\vec{x}_i)_s^{\perp}$ and set $\vec{\nu} = (\vec{\nu}_1, \dots, \vec{\nu}_{N_C})$. We further make use of the shorthand notation

$$(\nabla_s u \,.\, \nabla_s v)_{|\Gamma_i} := (u_s \,.\, v_s)_{|\Gamma_i} := (u_i)_s \,.\, (v_i)_s = \frac{(u_i)_\rho \,.\, (v_i)_\rho}{\|(\vec{x}_i)_\rho\|^2}.$$

Further we use the shorthand notation $F = (F_1, \dots, F_{N_C})$ with F_i given by (3.34).

Now, we consider time-dependent curves. A possible weak scheme is now given as follows: Find time-dependent functions $\vec{x}$ and κ, with $\vec{x}(\,.\,, t) \in \underline{V}$, $\vec{x}_t(\,.\,, t) \in \underline{V}_\partial$ and $\kappa(\,.\,, t) \in W$ for $t \in [0, T]$, such that

$$\int_{\Gamma(t)} \vec{x}_t \,.\, \vec{\nu}\, \chi \,\mathrm{d}s - \sigma \int_{\Gamma(t)} \kappa\, \chi \,\mathrm{d}s = \int_{\Gamma(t)} F\, \chi \,\mathrm{d}s, \qquad \forall \chi \in W, \tag{3.39a}$$

$$\int_{\Gamma(t)} \kappa\, \vec{\nu} \,.\, \vec{\eta} \,\mathrm{d}s + \int_{\Gamma(t)} \vec{x}_s \,.\, \vec{\eta}_s \,\mathrm{d}s = 0, \qquad \forall \vec{\eta} \in \underline{V}_\partial. \tag{3.39b}$$

Proof. Let $\vec{x}$ and κ solve the weak scheme (3.39) and let $\vec{x}_{ss}$ exist a.e. The strong scheme (3.35), (3.36) and (3.37) can be derived from the weak scheme and the regularity assumption:

From (3.39a), we get $(\vec{x}_i)_t \,.\, \vec{\nu}_i = \sigma\kappa_i + F_i$ a.e., for each $i \in \{1, \dots, N_C\}$. With integration by parts, equation (3.39b) can be reformulated to

$$\int_{\Gamma(t)} (\kappa\,\vec{\nu} - \vec{x}_{ss}) \,.\, \vec{\eta}\,\mathrm{d}s + \sum_{\substack{i\in\{1,\dots,N_C\},\\ \partial\Gamma_i(t)\neq\emptyset}} [(\vec{x}_i)_s(1) \,.\, \vec{\eta}_i(1) - (\vec{x}_i)_s(0) \,.\, \vec{\eta}_i(0)] = 0, \qquad \forall \vec{\eta} \in \underline{V}_\partial.$$

Choose $\vec{\eta} \in \underline{V}_\partial$ with $\vec{\eta}_i(0) = \vec{\eta}_i(1) = 0$ for $i \in \{1, \dots, N_C\}$ with $\partial\Gamma_i(t) \neq \emptyset$. This leads to $\kappa_i\,\vec{\nu}_i = (\vec{x}_i)_{ss}$ a.e. for each $i \in \{1, \dots, N_C\}$. The equation above reduces to

$$\sum_{\substack{i\in\{1,\dots,N_C\},\\ \partial\Gamma_i(t)\neq\emptyset}} [(\vec{x}_i)_s(1) \,.\, \vec{\eta}_i(1) - (\vec{x}_i)_s(0) \,.\, \vec{\eta}_i(0)] = 0, \qquad \forall \vec{\eta} \in \underline{V}_\partial. \tag{3.40}$$

Consider a triple point $\vec{\Lambda}_k$, $k \in \{1, \dots, N_T\}$. The attachment condition $\vec{x}_{i_{k,1}}(\rho_{k,1}) = \vec{x}_{i_{k,2}}(\rho_{k,2}) = \vec{x}_{i_{k,3}}(\rho_{k,3}) = \vec{\Lambda}_k$ is satisfied by $\vec{x} \in \underline{V}$. Choose a test function $\vec{\eta} \in \underline{V}_\partial$ with $\vec{\eta}_i(0) = \vec{\eta}_i(1) = 0$ for $\partial\Gamma_i(t) \neq \emptyset$, $i \notin \{i_{k,1}, i_{k,2}, i_{k,3}\}$ and $\vec{\eta}_{i_{k,l}}(1 - \rho_{k,l}) = 0$ for $l = 1, 2, 3$. Since $\vec{\eta}_0 := \vec{\eta}_{i_{k,1}}(\rho_{k,1}) = \vec{\eta}_{i_{k,2}}(\rho_{k,2}) = \vec{\eta}_{i_{k,3}}(\rho_{k,3}) \in \mathbb{R}^2$ can be arbitrary chosen, we conclude from (3.40)

$$\sum_{l=1}^{3} (-1)^{\rho_{k,l}} (\vec{x}_{i_{k,l}})_s(\rho_{k,l}) = 0,$$

which is Young's law (3.36b).

Consider a boundary intersection point $\vec{Q}_k$, $k \in \{1, \dots, N_I\}$. Since $\vec{x}_t \in \underline{V}_\partial$, the attachment condition $(\vec{x}_{i_{I,k}})_t(\rho_{I,k}) \,.\, \vec{n}_{\partial\Omega}(\vec{Q}_k) = 0$ is satisfied, i.e. the curve endpoint at $\partial\Omega$ can only move in tangential direction along $\partial\Omega$. Choose a test function $\vec{\eta} \in \underline{V}_\partial$ with with $\vec{\eta}_i(0) = \vec{\eta}_i(1) = 0$ for $\partial\Gamma_i(t) \neq \emptyset$, $i \neq i_{I,k}$ and $\vec{\eta}_{i_{I,k}}(1 - \rho_{I,k}) = 0$. Set $\vec{\eta}_0 := \vec{\eta}_{i_{I,k}}(\rho_{I,k})$. Equation (3.40) reduces to

$$(\vec{x}_{i_{I,k}})_s(\rho_{I,k}) \,.\, \vec{\eta}_0 = 0.$$

Since $\vec{\eta}$ is an element in $\underline{V}_\partial$, the vectors $\vec{\eta}_0$ and $\vec{n}_{\partial\Omega}(\vec{Q}_k)$ are orthogonal. Consequently, the tangential vector $(\vec{x}_{i_{I,k}})_s(\rho_{I,k})$ is parallel to $\vec{n}_{\partial\Omega}(\vec{Q}_k)$. Thus, the curve $\Gamma_{i_{I,k}}(t)$ meets the external boundary $\partial\Omega$ at $\vec{Q}_k$ with an angle of 90 degrees.

Similarly, we can derive the weak scheme from the strong scheme by using integration by parts where the boundary integrals diminish due to the conditions at triple junctions and boundary intersection points. □

3.2.8 Image Restoration with Edge Enhancement

The presented image segmentation method automatically provides a piecewise constant approximation of a possibly noisy image given by (3.19). For a variety of real images, a piecewise constant approximation poses a too large simplification. A piecewise smooth approximation can be found by reconsidering the Mumford-Shah functional (3.16), with approximations $u_{|\Omega_k} = u_k$, $k = 1, \dots, N_R$, where $u_k \in C^1(\Omega_k, \mathbb{R})$. Fixing curves $\Gamma_1, \dots, \Gamma_{N_C}$, we consider the following energy functional:

$$E^{MS,2}(u_1, \dots, u_{N_R}) = \sum_{k=1}^{N_R} \left(\lambda_k \int_{\Omega_k} (u_0 - u_k)^2 \,\mathrm{d}x + \int_{\Omega_k} \|\nabla u_k\|^2 \,\mathrm{d}x \right). \tag{3.41}$$

In contrast to (3.16), we also allow region-dependent parameters $\lambda_k > 0$.

By considering variations of the form $u_k + \epsilon\eta$, $\epsilon \in (-\epsilon_0, \epsilon_0)$, $\eta \in C^1(\overline{\Omega_k}, \mathbb{R})$, in (3.41), we obtain, using the classical methods from the theory of calculus of variations, an elliptic, boundary value problem. In detail, we compute

$$\begin{aligned}\delta E_k^{MS,2}(\eta) &:= \frac{\mathrm{d}}{\mathrm{d}\epsilon}\bigg|_{\epsilon=0} E^{MS,2}(u_1, \ldots, u_k + \epsilon\eta, \ldots, u_{N_R}) \\ &= \frac{\mathrm{d}}{\mathrm{d}\epsilon}\bigg|_{\epsilon=0} \left(\lambda_k \int_{\Omega_k} (u_0 - u_k - \epsilon\eta)^2 \,\mathrm{d}x + \int_{\Omega_k} \|\nabla u_k + \epsilon\nabla\eta\|^2 \,\mathrm{d}x\right) \\ &= 2\lambda_k \int_{\Omega_k} (u_k - u_0)\eta \,\mathrm{d}x + 2\int_{\Omega_k} \nabla u_k \,.\, \nabla\eta \,\mathrm{d}x.\end{aligned}$$

A necessary condition for a minimum is $\delta E_k^{MS,2}(\eta) = 0$ for all η. Therefore, we consider the equation

$$0 = \int_{\Omega_k} (\lambda_k(u_k - u_0)\eta + \nabla u_k \,.\, \nabla\eta) \,\mathrm{d}x.$$

For $u_k \in C^2(\Omega_k, \mathbb{R}) \cap C^1(\overline{\Omega_k}, \mathbb{R})$ integration by parts can be performed, which leads to

$$0 = \int_{\Omega_k} (\lambda_k(u_k - u_0) - \Delta u_k)\,\eta \,\mathrm{d}x + \int_{\partial\Omega_k} \nabla u_k \,.\, \vec{n}_{\partial\Omega_k}\eta \,\mathrm{d}s, \tag{3.42}$$

where $\vec{n}_{\partial\Omega_k}$ is the unit outer normal vector field on $\partial\Omega_k$. Choosing $\eta = 0$ on $\partial\Omega_k$ and η arbitrary inside of Ω_k, we get the partial differential equation

$$-\frac{1}{\lambda_k}\Delta u_k + u_k = u_0 \quad \text{in } \Omega_k. \tag{3.43a}$$

Thus, (3.42) reduces to

$$0 = \int_{\partial\Omega_k} \nabla u_k \,.\, \vec{n}_{\partial\Omega_k}\eta \,\mathrm{d}s,$$

from which we can conclude the Neumann boundary condition

$$\nabla u_k \,.\, \vec{n}_{\partial\Omega_k} = 0 \quad \text{on } \partial\Omega_k. \tag{3.43b}$$

We separately solve the partial differential equation with Neumann boundary conditions in the regions $\Omega_1, \ldots, \Omega_{N_R}$. The solution u poses an approximation of the original image u_0. The parameter $1/\lambda_k > 0$ controls the smoothing effect of the Laplace operator. The bigger λ_k is, the closer is the approximation to the original image. The smaller λ_k is, the smoother is u. Since we solve the partial differential equations with Neumann boundary conditions separately in each region, the image is not smoothed out at the interfaces Γ_i. Consequently the edges of objects in the image will remain sharp if the interfaces match with the edges.

The system (3.43) is close to the scheme proposed by Tsai et al. (2001), where the authors derived a level set method for two phases from the Mumford-Shah functional by considering the approximation u as the optimal estimate of a stochastic process.

In contrast to Hintermüller and Ring (2004), Chung and Vese (2009), and Tsai et al. (2001), who use a piecewise smooth approximating function in the segmentation step, we do not solve the diffusion scheme (3.43) in each time step of the curve evolution. Instead, we use the piecewise constant model to achieve a fast image segmentation followed by a smoothing of the image as a postprocessing step.

3.2.9 Color Images

In this section, we state how the methods developed for gray-scaled images can be generalized for color images. For color images, the image denoising scheme (3.43) is solved for each RGB color channel individually. For image segmentation we have to solve the scheme (3.35) similar as in the scalar case. Only the external forcing term F_i is adapted for color images.

RGB (red-green-blue) color space

A color image is given by a vector-valued image function $\vec{u}_0 = (u_{0,1}, u_{0,2}, u_{0,3}) : \Omega \to [0,1]^3$, where the components of $\vec{u}_0$ denote the red, green, and blue (RGB) color channel of the image. Using the RGB image data, we consider the following energy (cf. Chan et al. (2000)):

$$E(\Gamma, \vec{u}) = \sigma|\Gamma| + \sum_{j=1}^{3} \lambda_j \int_\Omega (u_{0,j} - u_j)^2 \,\mathrm{d}x, \tag{3.44}$$

where $\vec{u} = (u_1, u_2, u_3) : \Omega \to \mathbb{R}^3$ is piecewise constant; i.e. $\vec{u}_{|\Omega_k} =: \vec{c}_k = (c_{k,1}, c_{k,2}, c_{k,3})$, $k = 1, \ldots, N_R$. Considering variations in the $c_{k,j}$ leads to

$$c_{k,j} = \frac{\int_{\Omega_k} u_{0,j} \,\mathrm{d}x}{\int_{\Omega_k} 1 \,\mathrm{d}x}, \quad j = 1,2,3,\ k = 1, \ldots, N_R. \tag{3.45}$$

Thus, each component $c_{k,j}$ of $\vec{c}_k$ is set to the mean of $u_{0,j}$ in Ω_k. Considering a variation in Γ leads to the evolution equation

$$(V_n)_i = \sigma\kappa_i + F_i, \quad i = 1, \ldots, N_C, \tag{3.46a}$$

$$F_i = \sum_{j=1}^{3} \lambda_j \left[(u_{0,j} - c_{k^+(i),j})^2 - (u_{0,j} - c_{k^-(i),j})^2 \right]. \tag{3.46b}$$

By modifying the weighting parameters λ_j, $j = 1,2,3$, we can control the segmentation with respect to the red, green and blue part of the image.

An overview on models for color image segmentation and restoration using the RGB space and generalizations of the Mumford-Shah functional is given by Brook et al. (2003).

Also for other vector-valued images $u_0 = (u_{0,1}, \ldots, u_{0,n}) : \Omega \to [0,1]^n$, where each component $u_{0,j}$ represents a different channel, a scheme like (3.46) can be used. In remote sensing, for instance, imaging sensors can have many different channels representing different spectral bands. For example, the Earth observation satellites LANDSAT-4 and LANDSAT-5 use seven spectral channels. Besides blue, green and red channels, four different infrared channels exist, see Markham and Barker (1985).

For some images, one may make use of alternative color spaces which allow one to consider a color's chromaticity, brightness, and saturation components separately. Two possible spaces are the CB (chromaticity-brightness) space and the HSV (hue, saturation and value) color space. The color spaces CB and HSV can be used for a variety of image processing tasks such as segmentation, denoising, or enhancement of color images, see Chan et al. (2001), Tang et al. (2002), Aujol and Kang (2006).

CB (chromaticity-brightness) color space

For an RGB image function $\vec{u}_0$, the chromaticity function $\vec{v}_0 : \Omega \to S^2 \subset \mathbb{R}^3$ and the brightness function $b_0 : \Omega \to \mathbb{R}$ are defined by

$$\vec{v}_0 = \frac{\vec{u}_0}{\|\vec{u}_0\|},\ b_0 = \|\vec{u}_0\|. \tag{3.47}$$

The energy we want to minimize is

$$E(\Gamma, \vec{v}, b) = \sigma|\Gamma| + \lambda_C \int_\Omega \|\vec{v}_0 - \vec{v}\|^2 \,\mathrm{d}x + \lambda_B \int_\Omega (b_0 - b)^2 \,\mathrm{d}x, \tag{3.48}$$

where $\vec{v} : \Omega \to \mathbb{R}^3$, $b : \Omega \to \mathbb{R}$ are piecewise constant; i.e. $\vec{v}_{|\Omega_k} =: \vec{v}_k \in \mathbb{R}^3$, $b_{|\Omega_k} =: b_k \in \mathbb{R}$, $k = 1, \ldots, N_R$. The parameters $\sigma, \lambda_C, \lambda_B > 0$ weight the length, chromaticity, and brightness terms. Further, the constraint $\|\vec{v}\| = 1$ needs to be satisfied. The constraint can be rewritten to

$$0 = g(\vec{v}) := \|\vec{v}\|^2 - 1. \tag{3.49}$$

Considering variations in b_k leads to

$$b_k = \frac{\int_{\Omega_k} b_0 \,\mathrm{d}x}{\int_{\Omega_k} 1 \,\mathrm{d}x}; \tag{3.50}$$

i.e. $b_k \in \mathbb{R}$ is set to the mean of the brightness $b_0 = \|\vec{u}_0\|$ in Ω_k.

We now fix Γ, b and consider the reduced functional

$$E_v(\vec{v}_1, \ldots, \vec{v}_{N_R}) = \lambda_C \sum_{k=1}^{N_R} \int_{\Omega_k} \|\vec{v}_k - \vec{v}_0\|^2 \,\mathrm{d}x \tag{3.51}$$

The functional E_v should be minimized w.r.t. $\vec{v}_k$, $k \in \{1, \ldots, N_R\}$, subject to (3.49). Therefore, consider a variation $\vec{v}_k + \epsilon\vec{\eta}$, $\vec{\eta} \in \mathbb{R}^3$, and compute

$$\begin{aligned}
\delta(E_v)_{\vec{v}_k}(\vec{\eta}) :=& \left.\frac{\mathrm{d}}{\mathrm{d}\epsilon}\right|_{\epsilon=0} E_v(\vec{v}_1, \ldots, \vec{v}_k + \epsilon\vec{\eta}, \ldots, \vec{v}_{N_R}) \\
=& 2\lambda_C \int_{\Omega_k} (\vec{v}_k - \vec{v}_0) \,.\, \vec{\eta} \,\mathrm{d}x \\
=& 2\lambda_C \left(\vec{v}_k \,.\, \vec{\eta}\, |\Omega_k| - \left(\int_{\Omega_k} \vec{v}_0 \,\mathrm{d}x \right) \,.\, \vec{\eta} \right),
\end{aligned}$$

where $|\Omega_k|$ denotes the area of Ω_k. Further, consider

$$\delta g_{\vec{v}_k}(\vec{\eta}) := \left.\frac{\mathrm{d}}{\mathrm{d}\epsilon}\right|_{\epsilon=0} g(\vec{v}_k + \epsilon\eta) = 2\vec{v}_k \,.\, \vec{\eta}.$$

For the constrained minimization problem, we need to solve

$$\delta(E_v)_{\vec{v}_k}(\vec{\eta}) = \mu\, \delta g_{\vec{v}_k}(\vec{\eta}),$$

where $\mu \in \mathbb{R}$ is a Lagrange multiplier. This equation can be rewritten to

$$\left(2\lambda_C |\Omega_k| \vec{v}_k - 2\lambda_C \int_{\Omega_k} \vec{v}_0 \,\mathrm{d}x \right) \,.\, \vec{\eta} = 2\mu \vec{v}_k \,.\, \vec{\eta}.$$

Since $\vec{\eta}$ is arbitrary, we conclude

$$\vec{v}_k = \frac{\lambda_C}{\lambda_C|\Omega_k| - \mu} \int_{\Omega_k} \vec{v}_0 \,\mathrm{d}x. \tag{3.52}$$

The Lagrange multiplier μ needs to be chosen such that the constraint $\|\vec{v}_k\| = 1$ is satisfied. This leads to the following computation of $\vec{v}_k$:

$$\vec{V}_k = \int_{\Omega_k} \vec{v}_0 \,\mathrm{d}x, \quad \vec{v}_k = \frac{\vec{V}_k}{\|\vec{V}_k\|}. \tag{3.53}$$

Finally, fixing $\vec{v}$ and $\vec{b}$ and considering a variation of Γ leads to the evolution equation

$$(V_n)_i = \sigma\kappa_i + F_i, \quad i = 1, \dots, N_C, \tag{3.54a}$$

$$F_i = \lambda_C \left[\|\vec{v}_0 - \vec{v}_{k^+(i)}\|^2 - \|\vec{v}_0 - \vec{v}_{k^-(i)}\|^2\right] + \lambda_B \left[(b_0 - b_{k^+(i)})^2 - (b_0 - b_{k^-(i)})^2\right]. \tag{3.54b}$$

HSV (hue-saturation-value) color space

In the HSV space, a color is given by three components: The periodical hue component describes the chromaticity ranging from red to yellow, green, cyan, blue, magenta, and back to red. The saturation ranges from 0 to 1. A gray color (all three components of the corresponding RGB color are equal) has a saturation value of 0. A saturation of 1 is a maximum saturated color; i.e. it has one RGB value equal to 0 and is therefore a mixture of only two RGB basic colors. The value component describes the luminosity of the color and is set to the maximum of the red, green, or blue value.

Let $\vec{h}_0 : \Omega \to S^1 \subset \mathbb{R}^2$, $s_0 : \Omega \to [0,1]$, and $v_0 : \Omega \to [0,1]$ be the HSV components corresponding to an RGB image function $\vec{u}_0$. The energy we want to minimize is

$$E(\Gamma, \vec{h}, s, v) = \sigma|\Gamma| + \lambda_H \int_\Omega \|\vec{h}_0 - \vec{h}\|^2 \mathrm{d}x + \lambda_S \int_\Omega (s_0 - s)^2 \mathrm{d}x + \lambda_V \int_\Omega (v_0 - v)^2 \mathrm{d}x, \tag{3.55}$$

where $\vec{h} : \Omega \to \mathbb{R}^2$, $s : \Omega \to [0,1]$, $v : \Omega \to [0,1]$ are piecewise constant; i.e. $\vec{h}_{|\Omega_k} =: \vec{h}_k \in \mathbb{R}^2$, $s_{|\Omega_k} =: s_k \in \mathbb{R}$, $v_{|\Omega_k} =: v_k \in \mathbb{R}$. The constraint $\|\vec{h}\| = 1$ needs to be satisfied. The parameters $\sigma, \lambda_H, \lambda_S, \lambda_V > 0$ weight the length, hue, saturation, and value terms. Similar as for the CB space (cf. (3.50), (3.53)), considering variations of the single components $\vec{h}_k$, s_k, v_k leads to

$$\vec{H}_k = \int_{\Omega_k} \vec{h}_0 \,\mathrm{d}x, \quad \vec{h}_k = \frac{\vec{H}_k}{\|\vec{H}_k\|}, \quad s_k = \frac{\int_{\Omega_k} s_0 \,\mathrm{d}x}{\int_{\Omega_k} 1 \,\mathrm{d}x}, \quad v_k = \frac{\int_{\Omega_k} v_0 \,\mathrm{d}x}{\int_{\Omega_k} 1 \,\mathrm{d}x}. \tag{3.56}$$

Here, a constrained minimization problem has to be solved for $\vec{h}_k$.

Fixing $\vec{h}$, s and v, and considering a variation of Γ leads to the evolution equation

$$(V_n)_i = \sigma\kappa_i + F_i, \quad i = 1, \dots, N_C, \tag{3.57a}$$

$$\begin{aligned} F_i = {} & \lambda_H \left[\|\vec{h}_0 - \vec{h}_{k^+(i)}\|^2 - \|\vec{h}_0 - \vec{h}_{k^-(i)}\|^2\right] + \lambda_S \left[(s_0 - s_{k^+(i)})^2 - (s_0 - s_{k^-(i)})^2\right] \\ & + \lambda_V \left[(v_0 - v_{k^+(i)})^2 - (v_0 - v_{k^-(i)})^2\right]. \end{aligned} \tag{3.57b}$$

To sum up, only the external forcing term F_i needs to be adapted for color images. The parametric scheme (3.35) can be used for both scalar and vector-valued images.

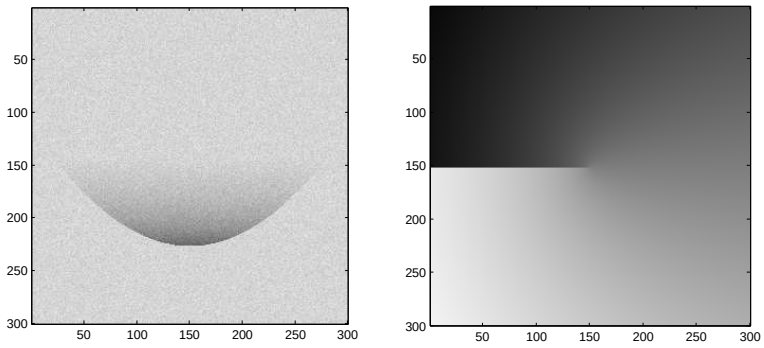


Figure 3.5: Images containing edges with free endpoints. Left: Image with an edge with two free endpoints. Right: Image with an edge with one free endpoint and one endpoint on the image boundary.

3.2.10 Contours with Free Endpoints

Up to now we restricted on curves which are interfaces between different regions. In the general Mumford-Shah problem (3.16) curves can be closed and open, with endpoints meeting at triple junctions, meeting the image boundary or being free endpoints (so-called *crack tips*, cf. Mumford and Shah (1989)). Two example images which contain edges with free endpoints are presented by Figure 3.5. The right image shows a crack tip problem which has also been studied by Pock et al. (2009b).

Open active contours, i.e. active contours with free endpoints, are also considered by Kimmel and Bruckstein (2003), where the authors propose an edge-based method for detection of open boundaries. Here, we consider approaches based on the Mumford-Shah model. Contours with free endpoints are also considered by Pock et al. (2009b) which propose a convex relaxation of the Mumford-Shah functional.

Fixing a curve Γ, let u denote the minimizer of the Mumford-Shah energy (3.16). At free endpoints, problems concerning the regularity of u occur, cf. Mumford and Shah (1989). Expressed in polar coordinates (r, ϕ) centered at the free endpoint, the solution u is of the form

$$u(r, \phi) = c\, r^{1/2} \sin(\frac{1}{2}(\phi - \phi_0)) + \hat{v}(r, \phi), \tag{3.58}$$

where $\hat{v}$ is a C^1-function and c, ϕ_0 are constants, see Aubert and Kornprobst (2006).

For image segmentation, we later need to solve the problem on a discrete set: Let Ω^h be a rectangular grid of nodes points covering Ω with grid size $h > 0$. We replace the second integral on the right hand side of (3.16) by a sum containing difference quotients of the form

$$\nabla_h^i u(\vec{z}) = \frac{1}{h}(u(\vec{z} + h\vec{e}_i) - u(\vec{z})), \quad \vec{z} \in \Omega^h, \quad i = 1, 2, \tag{3.59}$$

where $\vec{e}_i \in \mathbb{R}^2$ are the standard basis vectors of $\mathbb{R}^2$. For image segmentation applications, we choose the pixel grid, i.e. we use $h = 1$. For the approximating sum, we have to exclude terms where the line $[\vec{z}, \vec{z} + h\vec{e}_i]$ intersects with the curve Γ.

Instead of the original Mumford-Shah functional (3.16), we thus consider the energy

$$\begin{aligned} E^h(\Gamma, u) =& \sigma|\Gamma| + \mu \sum_{\vec{z} \in \Omega^h} \left((1 - \alpha_x(\vec{z}))(\nabla_h^2 u(\vec{z}))^2 + (1 - \alpha_y(\vec{z}))(\nabla_h^1 u(\vec{z}))^2\right) \\ &+ \lambda \int_\Omega (u_0 - u)^2 \, \mathrm{d}x, \end{aligned} \tag{3.60}$$

where $\alpha_x(\vec{z}), \alpha_y(\vec{z}) \in [0,1]$ are scalar terms. For example, if $[\vec{z}, \vec{z} + h\vec{e}_1]$ intersects with Γ, $\alpha_y(\vec{z})$ is set to 1. The detailed definitions are given below. Further, the energy (3.60) contains an additional weighting parameter $\mu > 0$. Note, that a third parameter μ is not necessary in general, since we could divide the energy by μ. This would result in a weight of $\mu = 1$ of the second term, and σ and λ would be replaced by σ/μ and λ/μ, respectively. As we will see in the following example, it is just necessary to set σ/μ sufficiently small to allow a growth of the curve in tangential direction at the free endpoints. In the images that we consider in the result section of this chapter, we often set σ to 1 and we set μ to a very large value. Alternatively, we could set μ to 1 and σ (and λ) to a corresponding small value.

Example

We consider one single open curve Γ. Let $\vec{x} : [0,1] \to \mathbb{R}^2$ with $\vec{x}([0,1]) = \Gamma$ be a parameterization of the curve. Let $\vec{x}(0)$ be a free endpoint and let $\vec{x}(1)$ intersect with the image boundary.

Figure 3.6 visualizes a possible situation near the free endpoint $\vec{x}(0)$. Let $\vec{z}_{++}$, $\vec{z}_{+-}$, $\vec{z}_{--}$, $\vec{z}_{-+}$ denote the four grid points around $\vec{x}(0)$ as shown in Figure 3.6. In this example, the tangential vector of the curve at $\vec{x}(0)$ is $\vec{\tau}(0) = \vec{x}_s(0) = \vec{e}_1$.

Considering $\vec{z} = \vec{z}_{+-}$, the line $[\vec{z}_{+-}, \vec{z}_{++}]$ and Γ intersect. Thus, $\alpha_x(\vec{z}_{+-})$ is set to 1. For $\vec{z} = \vec{z}_{--}$, we define a factor

$$\alpha_x(\vec{z}_{--}) := \frac{1}{h}\left((\vec{z}_{+-})_1 - (\vec{x}(0))_1\right), \tag{3.61}$$

where $(\,.\,)_i$ denotes the i-th component of a vector, $i = 1, 2$. The factor $\alpha_x(\vec{z}_{--})$ describes how far the curve has entered the square given by $\vec{z}_{++}$, $\vec{z}_{+-}$, $\vec{z}_{--}$, $\vec{z}_{-+}$.

For $\vec{z} \in \Omega^h$, $\vec{z} \neq \vec{z}_{--}$, we set $\alpha_x(\vec{z}) = 0$ if $[\vec{z}, \vec{z}+h\vec{e}_2] \cap \Gamma = \emptyset$ and $\alpha_x(\vec{z}) = 1$ else. The factor $\alpha_y(\vec{z})$ is defined similarly; since $\vec{\tau}(0) = \vec{e}_1$ in this example, $\alpha_y(\vec{z}) = 0$ if $[\vec{z}, \vec{z} + h\vec{e}_1] \cap \Gamma = \emptyset$ and 1 else.

Let Γ grow in direction $-\vec{\tau}(0)$ at $\vec{x}(0)$. We consider a second curve Γ^ϵ, $\epsilon > 0$, with a parameterization $\vec{x}^\epsilon$, such that $\vec{x}^\epsilon(0) = \vec{x}(0) - \epsilon\vec{\tau}(0)$. We can assume that ϵ is small enough, such that $\vec{x}^\epsilon(0)$ is still inside the square given by $\vec{z}_{++}$, $\vec{z}_{+-}$, $\vec{z}_{--}$, $\vec{z}_{-+}$.

The energy difference is

$$\begin{aligned} E^h(\Gamma^\epsilon, u) - E^h(\Gamma, u) =& \sigma\epsilon + \mu(1 - \alpha_x(\vec{z}_{--}) - \epsilon)(\nabla_h^2 u(\vec{z}_{--}))^2 - \mu(1 - \alpha_x(\vec{z}_{--}))(\nabla_h^2 u(\vec{z}_{--}))^2 \\ =& \sigma\epsilon - \epsilon\mu(\nabla_h^2 u(\vec{z}_{--}))^2. \end{aligned}$$

The energy will decrease if

$$\sigma < \mu(\nabla_h^2 u(\vec{z}_{--}))^2. \tag{3.62}$$

Thus a motion of a curve in $-\vec{\tau}(0) = -\vec{e}_1$ direction at the free endpoint $\vec{x}(0)$ requires that the square of the difference quotient of u in $\vec{e}_2$-direction at $\vec{z}_{--}$ weighted with μ is sufficient large compared to σ.

General case

We consider a curve with two free endpoints with arbitrary tangential vectors $\vec{\tau}(0)$ and $\vec{\tau}(1)$ at the free endpoints. We define the factors α_x and α_y as follows: Let $\vec{x}(\rho)$, $\rho \in \{0,1\}$ be a free endpoint and $\vec{z}_{++}(\rho), \vec{z}_{+-}(\rho), \vec{z}_{--}(\rho), \vec{z}_{-+}(\rho)$ denote the four grid points around $\vec{x}(\rho)$.

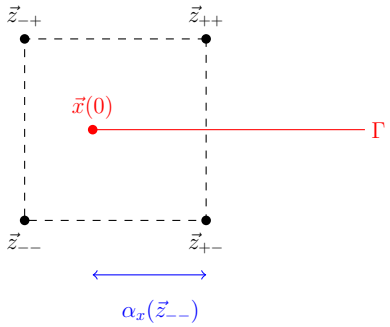


Figure 3.6: Illustration of the pixel grid close to the free endpoint.

If $\vec{\tau}(0) \,.\, \vec{e}_1 \geq 0$, we define:

$$\vec{z}^{\,0,1} := \vec{z}_{--}(0), \quad \alpha_x(\vec{z}^{\,0,1}) := 1 - \frac{1}{h}\left((\vec{x}(0))_1 - (\vec{z}^{\,0,1})_1\right).$$

Otherwise, i.e. if $\vec{\tau}(0) \,.\, \vec{e}_1 < 0$, we define:

$$\vec{z}^{\,0,1} := \vec{z}_{+-}(0), \quad \alpha_x(\vec{z}^{\,0,1}) := 1 - \frac{1}{h}\left((\vec{z}^{\,0,1})_1 - (\vec{x}(0))_1\right).$$

If $\vec{\tau}(0) \,.\, \vec{e}_2 \geq 0$, we define:

$$\vec{z}^{\,0,2} := \vec{z}_{--}(0), \quad \alpha_y(\vec{z}^{\,0,2}) := 1 - \frac{1}{h}\left((\vec{x}(0))_2 - (\vec{z}^{\,0,2})_2\right).$$

Otherwise, i.e. if $\vec{\tau}(0) \,.\, \vec{e}_2 < 0$, we define:

$$\vec{z}^{\,0,2} := \vec{z}_{-+}(0), \quad \alpha_y(\vec{z}^{\,0,2}) := 1 - \frac{1}{h}\left((\vec{z}^{\,0,2})_2 - (\vec{x}(0))_2\right).$$

If $\vec{\tau}(1) \,.\, \vec{e}_1 \geq 0$, we define:

$$\vec{z}^{\,1,1} := \vec{z}_{+-}(1), \quad \alpha_x(\vec{z}^{\,1,1}) := 1 - \frac{1}{h}\left((\vec{z}^{\,1,1})_1 - (\vec{x}(1))_1\right).$$

Otherwise, i.e. if $\vec{\tau}(1) \,.\, \vec{e}_1 < 0$, we define:

$$\vec{z}^{\,1,1} := \vec{z}_{--}(1), \quad \alpha_x(\vec{z}^{\,1,1}) := 1 - \frac{1}{h}\left((\vec{x}(1))_1 - (\vec{z}^{\,1,1})_1\right).$$

If $\vec{\tau}(1) \,.\, \vec{e}_2 \geq 0$, we define:

$$\vec{z}^{\,1,2} := \vec{z}_{-+}(1), \quad \alpha_y(\vec{z}^{\,1,2}) := 1 - \frac{1}{h}\left((\vec{z}^{\,1,2})_2 - (\vec{x}(1))_2\right).$$

Otherwise, i.e. if $\vec{\tau}(1) \,.\, \vec{e}_2 < 0$, we define:

$$\vec{z}^{\,1,2} := \vec{z}_{--}(1), \quad \alpha_y(\vec{z}^{\,1,2}) := 1 - \frac{1}{h}\left((\vec{x}(1))_2 - (\vec{z}^{\,1,2})_2\right).$$

We define the following factors for $\vec{z} \in \Omega^h$:

$$\alpha_x(\vec{z}) = \begin{cases} 1, & \text{if } [\vec{z}, \vec{z} + h\vec{e}_2] \cap \Gamma \neq \emptyset,\ \vec{z} \neq \vec{z}^{0,1},\ \vec{z} \neq \vec{z}^{1,1}, \\ \alpha_x(\vec{z}^{\rho,1}), & \text{if } \vec{z} = \vec{z}^{\rho,1}, \rho \in \{0,1\}, \\ 0, & \text{else.} \end{cases}$$

and

$$\alpha_y(\vec{z}) = \begin{cases} 1, & \text{if } [\vec{z}, \vec{z} + h\vec{e}_1] \cap \Gamma \neq \emptyset,\ \vec{z} \neq \vec{z}^{0,2},\ \vec{z} \neq \vec{z}^{1,2}, \\ \alpha_y(\vec{z}^{\rho,2}), & \text{if } \vec{z} = \vec{z}^{\rho,2}, \rho \in \{0,1\}, \\ 0, & \text{else.} \end{cases}$$

Evolution equations

We consider the energy (3.60). First, we fix u and consider for $\vec{\eta} : [0,1] \to \mathbb{R}^2$ a variation of $\vec{x}$ of the form $\vec{x} + \epsilon\vec{\eta}$, $\epsilon > 0$.

Before we proceed computing the change in the energy, we shortly consider the changes in α_x and α_y. Consider $\rho = 1$ and the case $\vec{\tau}(1) \,.\, \vec{e}_1 \geq 0$. For $\vec{x} + \epsilon\vec{\eta}$, we obtain:

$$\alpha_x^\epsilon(\vec{z}^{1,1}) := 1 - \frac{1}{h}\left((\vec{z}^{1,1})_1 - (\vec{x}(1))_1 - \epsilon\vec{\eta}(1) \,.\, \vec{e}_1\right).$$

To compute later the resulting difference in the energy, we need to compute

$$(1 - \alpha_x^\epsilon(\vec{z}^{1,1})) - (1 - \alpha_x(\vec{z}^{1,1})) = -\epsilon\,\vec{\eta}(1) \,.\, \vec{e}_1.$$

Similarly, for $\vec{\tau}(1) \,.\, \vec{e}_1 < 0$, we obtain

$$(1 - \alpha_x^\epsilon(\vec{z}^{1,1})) - (1 - \alpha_x(\vec{z}^{1,1})) = \epsilon\,\vec{\eta}(1) \,.\, \vec{e}_1.$$

We can summarize the two cases to

$$(1 - \alpha_x^\epsilon(\vec{z}^{1,1})) - (1 - \alpha_x(\vec{z}^{1,1})) = -\text{sign}(\vec{\tau}(1) \,.\, \vec{e}_1)\,\epsilon\,\vec{\eta}(1) \,.\, \vec{e}_1.$$

Similar, we compute

$$\begin{aligned} (1 - \alpha_y^\epsilon(\vec{z}^{1,2})) - (1 - \alpha_y(\vec{z}^{1,2})) &= -\text{sign}(\vec{\tau}(1) \,.\, \vec{e}_2)\,\epsilon\,\vec{\eta}(1) \,.\, \vec{e}_2, \\ (1 - \alpha_x^\epsilon(\vec{z}^{0,1})) - (1 - \alpha_x(\vec{z}^{0,1})) &= +\text{sign}(\vec{\tau}(0) \,.\, \vec{e}_1)\,\epsilon\,\vec{\eta}(0) \,.\, \vec{e}_1, \\ (1 - \alpha_y^\epsilon(\vec{z}^{0,2})) - (1 - \alpha_y(\vec{z}^{0,2})) &= +\text{sign}(\vec{\tau}(0) \,.\, \vec{e}_2)\,\epsilon\,\vec{\eta}(0) \,.\, \vec{e}_2. \end{aligned}$$

Let $\Gamma^{\epsilon,\eta}$ denote the image of $\vec{x} + \epsilon\vec{\eta}$. We use the notation $E^h(\vec{\eta}) := E^h(\Gamma^{\epsilon,\eta}, u)$ and compute

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}\epsilon}\bigg|_{\epsilon=0} E^h(\vec{\eta}) &= \frac{\mathrm{d}}{\mathrm{d}\epsilon}\bigg|_{\epsilon=0} E^h(\Gamma^{\epsilon,\eta}, u) \\ &= -\sigma \int_\Gamma \vec{x}_{ss} \,.\, \vec{\eta}\,\mathrm{d}s - \int_\Gamma F\,\vec{\nu} \,.\, \vec{\eta}\,\mathrm{d}s + \sigma\vec{x}_s(1) \,.\, \vec{\eta}(1) - \sigma\vec{x}_s(0) \,.\, \vec{\eta}(0) \\ &\quad - \mu\left[\text{sign}(\vec{\tau}(1) \,.\, \vec{e}_1)\vec{\eta}(1) \,.\, \vec{e}_1(\nabla_h^2 u(\vec{z}^{1,1}))^2 + \text{sign}(\vec{\tau}(1) \,.\, \vec{e}_2)\vec{\eta}(1) \,.\, \vec{e}_2(\nabla_h^1 u(\vec{z}^{1,2}))^2\right] \\ &\quad + \mu\left[\text{sign}(\vec{\tau}(0) \,.\, \vec{e}_1)\vec{\eta}(0) \,.\, \vec{e}_1(\nabla_h^2 u(\vec{z}^{0,1}))^2 + \text{sign}(\vec{\tau}(0) \,.\, \vec{e}_2)\vec{\eta}(0) \,.\, \vec{e}_2(\nabla_h^1 u(\vec{z}^{0,2}))^2\right]. \end{aligned}$$

For this computation, integration by parts and a transport theorem are applied. Further F is defined as the jump

$$F = \lambda[(u_0 - u^+)^2 - (u_0 - u^-)^2] \tag{3.63}$$

with

$$u^{\pm}(\vec{y}) = \lim_{a \to 0} u(\vec{y} \pm a\vec{\nu}(\vec{y})) \tag{3.64}$$

for $\vec{y} \in \Gamma$.

We define the following inner product for functions $\vec{\eta}$, $\vec{\chi} : [0,1] \to \mathbb{R}^2$:

$$(\vec{\eta}, \vec{\chi})_{2,\Gamma,\partial\Gamma} := \int_{\Gamma} \vec{\eta} \,.\, \vec{\chi} \, \mathrm{d}s + \vec{\eta}(1) \,.\, \vec{\chi}(1) + \vec{\eta}(0) \,.\, \vec{\chi}(0). \tag{3.65}$$

Now, we consider a family of curves $\Gamma(t)$, $t \in [0,T]$. Let $\vec{x} : [0,1] \times [0,T] \to \mathbb{R}^2$ be a mapping such that $\vec{x}(\,.\,,t)$ is a parameterization of $\Gamma(t)$, $t \in [0,T]$. We call $\vec{x}$ a solution of the gradient flow equation, if

$$(\vec{x}_t, \vec{\eta})_{2,\Gamma,\partial\Gamma} = -\left.\frac{\mathrm{d}}{\mathrm{d}\epsilon}\right|_{\epsilon=0} E^h(\vec{\eta}) \tag{3.66}$$

holds for all $\vec{\eta} : [0,1] \to \mathbb{R}^2$.

First, we consider $\vec{\eta} = \eta_0 \, \vec{\nu}$ for a scalar function $\eta_0 : [0,1] \to \mathbb{R}$ with $\eta_0(0) = \eta_0(1) = 0$. This provides

$$\int_{\Gamma(t)} \vec{x}_t \,.\, \vec{\nu} \, \eta_0 \, \mathrm{d}s = \int_{\Gamma(t)} (\sigma \vec{x}_{ss} \,.\, \vec{\nu} + F) \, \eta_0 \, \mathrm{d}s. \tag{3.67}$$

Since η_0 is arbitrary chosen (with value 0 at the endpoints), we conclude the following evolution equation:

$$\vec{x}_t \,.\, \vec{\nu} = \sigma\kappa + F. \tag{3.68}$$

Now choose $\vec{\eta} = \eta_0 \vec{\tau}$, where $\eta_0 : [0,1] \to \mathbb{R}$ is a scalar function with $\eta_0(0) \neq 0$ and $\eta_0(1) = 0$, i.e. $\vec{\eta}(0) = \vec{\tau}(0)\eta_0(0)$ and $\vec{\eta}(1) = \vec{0}$. Inserting $\vec{\eta}$ in (3.66) and using (3.68) and $\vec{\tau} = \vec{x}_s$, leads to

$$\begin{aligned} \vec{x}_t(0) \,.\, \vec{\tau}(0)\eta_0(0) =& \sigma\eta_0(0) - \mu \left[\mathrm{sign}(\vec{\tau}(0) \,.\, \vec{e}_1)\vec{\tau}(0)\eta_0(0) \,.\, \vec{e}_1 (\nabla_h^2 u(\vec{z}^{0,1}))^2 + \right. \\ & \left. + \, \mathrm{sign}(\vec{\tau}(0) \,.\, \vec{e}_2)\vec{\tau}(0)\eta_0(0) \,.\, \vec{e}_2 (\nabla_h^1 u(\vec{z}^{0,2}))^2 \right]. \end{aligned} \tag{3.69}$$

Since $\eta_0(0)$ is arbitrary and $\mathrm{sign}(\vec{\tau}(0) \,.\, \vec{e}_i)\vec{\tau}(0) \,.\, \vec{e}_i = |\vec{\tau}(0) \,.\, \vec{e}_i|$ for $i = 1, 2$, we conclude for the tangential velocity

$$V_{\tan}(0) := \vec{x}_t(0) \,.\, \vec{\tau}(0) = \sigma - \mu \left[|\vec{\tau}(0) \,.\, \vec{e}_1| (\nabla_h^2 u(\vec{z}^{0,1}))^2 + |\vec{\tau}(0) \,.\, \vec{e}_2| (\nabla_h^1 u(\vec{z}^{0,2}))^2 \right]. \tag{3.70}$$

The curve Γ will grow locally at $\vec{x}(0)$ in tangential direction $-\vec{\tau}(0)$, if $V_{\tan}(0) < 0$. Therefore,

$$\sigma < \mu \left[|\vec{\tau}(0) \,.\, \vec{e}_1| (\nabla_h^2 u(\vec{z}^{0,1}))^2 + |\vec{\tau}(0) \,.\, \vec{e}_2| (\nabla_h^1 u(\vec{z}^{0,2}))|^2 \right] \tag{3.71}$$

has to be satisfied such that the curve grows at $\vec{x}(0)$.

As an example, we consider the case $\vec{\tau}(0) = \vec{e}_1$. Equation (3.70) reduces to $V_{\tan}(0) = \sigma - \mu(\nabla_h^2 u(\vec{z}^{0,1}))^2$. For a growth of $\Gamma(t)$ in negative $\vec{e}_1$-direction, condition (3.62) from the introductory example has to be satisfied (replacing $\vec{z}_{--}$ by $\vec{z}^{0,1}$).

Similarly, choosing $\eta_0(0) = 0$ and $\eta_0(1) \neq 0$, we can derive the following equation for the tangential velocity in $\vec{x}(1)$:

$$V_{\tan}(1) := \vec{x}_t(1) \,.\, \vec{\tau}(1) = -\sigma + \mu \left[|\vec{\tau}(1) \,.\, \vec{e}_1| (\nabla_h^2 u(\vec{z}^{1,1}))^2 + |\vec{\tau}(1) \,.\, \vec{e}_2| (\nabla_h^1 u(\vec{z}^{1,2}))^2 \right]. \tag{3.72}$$

The curve $\Gamma(t)$ will grow in tangential direction $\vec{\tau}(1)$, if $V_{\tan}(1) > 0$. Therefore, the inequality

$$\sigma < \mu \left[|\vec{\tau}(1) \,.\, \vec{e}_1| (\nabla_h^2 u(\vec{z}^{1,1}))^2 + |\vec{\tau}(1) \,.\, \vec{e}_2| (\nabla_h^1 u(\vec{z}^{1,2}))^2 \right] \tag{3.73}$$

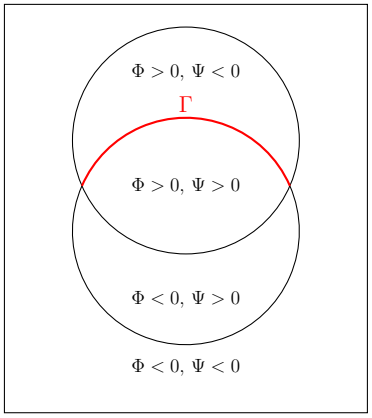


Figure 3.7: Illustration of the level set method to represent a curve Γ with free endpoints; two level set functions Φ and Ψ and auxiliary regions are needed.

has to be satisfied.

Since the term $\sigma|\Gamma|$ in the energy (3.60) penalizes the length of the curve, a curve can only grow in direction $-\vec{\tau}(0)$ or $\vec{\tau}(1)$ at the endpoints, if the derivative terms $(\nabla_h^i u)^2$ (weighted with μ), $i = 1, 2$, are large compared to σ.

Choosing $\vec{\eta} = \eta_0 \vec{\nu}$, provides the following equations for the normal velocity at the free endpoints:

$$\begin{aligned} V_{\mathrm{n}}(0) &:= \vec{x}_t(0) \,.\, \vec{\nu}(0) \\ &= -\mu \left[\operatorname{sign}(\vec{\tau}(0) \,.\, \vec{e}_1) \vec{\nu}(0) \,.\, \vec{e}_1 (\nabla_h^2 u(\vec{z}^{0,1}))^2 + \operatorname{sign}(\vec{\tau}(0) \,.\, \vec{e}_2) \vec{\nu}(0) \,.\, \vec{e}_2 (\nabla_h^1 u(\vec{z}^{0,2}))^2 \right], \end{aligned} \tag{3.74}$$

$$\begin{aligned} V_{\mathrm{n}}(1) &:= \vec{x}_t(1) \,.\, \vec{\nu}(1) \\ &= +\mu \left[\operatorname{sign}(\vec{\tau}(1) \,.\, \vec{e}_1) \vec{\nu}(1) \,.\, \vec{e}_1 (\nabla_h^2 u(\vec{z}^{1,1}))^2 + \operatorname{sign}(\vec{\tau}(1) \,.\, \vec{e}_2) \vec{\nu}(1) \,.\, \vec{e}_2 (\nabla_h^1 u(\vec{z}^{1,2}))^2 \right]. \end{aligned} \tag{3.75}$$

Diffusion equations

We consider the energy (3.60) and we now fix the curve Γ. The approximation u is defined by its values on the discrete set Ω^h. The scheme (3.43) has to be replaced by a discrete scheme, since we have replaced the gradient terms by difference quotients in (3.60). In Section 3.3.7, we will illustrate how such a discrete scheme for image diffusion can be solved.

Remark 3.2 (Comparison with Level Set Techniques)**.** *The easy handling of non-interface curves is an advantage of the parametric method. With level set methods free endpoints cannot be handled with one level set function, see Schaeffer and Vese (2014). Since level set methods embed a curve as zero level set of a function $\Phi : \Omega \to \mathbb{R}$, the curve is an interface between two regions $\{\Phi > 0\}$ and $\{\Phi < 0\}$. Therefore level sets are always closed or meet the image boundary at their endpoints. Non-interface curves can be handled by using two level set functions Φ and Ψ and by using artificial regions. A curve with free endpoints can then be represented by the interface between the artificial regions $\{\Phi > 0\} \cap \{\Psi > 0\}$ and $\{\Phi > 0\} \cap \{\Psi < 0\}$, for example, cf. Figure 3.7. Details on this method are given in Schaeffer and Vese (2014). Using our direct, parametric approach it is not necessary to introduce artificial regions.*

3.3 Numerical Approximation

3.3.1 Finite Element Approximation

We introduce a finite element approximation for the scheme (3.35) with (3.36) and (3.37) in the case of triple junctions and boundary intersections. The finite element scheme is based on the weak formulation (3.39).

The flow (3.35) can be interpreted as a weighted mean curvature flow with an external forcing term. Therefore, the approach in this work follows the ideas of Barrett et al. (2007a), who consider mean curvature flow and related flows. We extend their parametric approach to solve the image segmentation scheme (3.35) with possible topology changes.

Spatial and Time Discretization

Let N_C be the number of curves. For $i = 1, \dots, N_C$, let $0 = q_0^i < q_1^i < \dots < q_{N_i}^i = 1$ be a decomposition of the interval $I_i = [0,1]$. If Γ_i is a closed curve, we make use of the periodicity $N_i = 0$, $N_i + 1 = 1$, $-1 = N_i - 1$, etc.

Let N_T be the number of triple junctions and N_I the number of boundary intersections. We adopt the notation from Section 3.2: For $k = 1, \dots, N_T$, let $\vec{\Lambda}_k$ denote a triple point. Let $i_{k,l}$, $l = 1,2,3$, be the indices of curves and $\rho_{k,l} \in \{0,1\}$ such that $\vec{x}_{i_{k,1}}(\rho_{k,1}) = \vec{x}_{i_{k,2}}(\rho_{k,2}) = \vec{x}_{i_{k,3}}(\rho_{k,3}) = \vec{\Lambda}_k$. For $k = 1, \dots, N_I$, let $\vec{Q}_k \in \partial\Omega$ denote a boundary intersection point. Let $i_{I,k}$ be the index of a curve and $\rho_{I,k} \in \{0,1\}$ such that $\vec{x}_{i_{I,k}}(\rho_{I,k}) = \vec{Q}_k$.

We introduce the following discrete function spaces

$$W^h := \Big\{ (\eta_1, \dots, \eta_{N_C}) \in C(I_1, \mathbb{R}) \times \dots \times C(I_{N_C}, \mathbb{R}) \,:\, \eta_i|_{[q_{j-1}^i, q_j^i]} \text{ is linear}, \\ \forall i = 1, \dots, N_C,\, j = 1, \dots, N_i \Big\}, \tag{3.76a}$$

$$\underline{V}^h := \Big\{ (\vec{\eta}_1, \dots, \vec{\eta}_{N_C}) \in C(I_1, \mathbb{R}^2) \times \dots \times C(I_{N_C}, \mathbb{R}^2) \,:\, \vec{\eta}_{i_{k,1}}(\rho_{k,1}) = \\ \vec{\eta}_{i_{k,2}}(\rho_{k,2}) = \vec{\eta}_{i_{k,3}}(\rho_{k,3}),\, \forall k = 1, \dots, N_T, \quad \vec{\eta}_i|_{[q_{j-1}^i, q_j^i]} \text{ is linear}, \\ \forall i = 1, \dots, N_C,\, j = 1, \dots, N_i \Big\}. \tag{3.76b}$$

A basis of W^h is given by functions $\chi_{i,j} := ((\chi_{i,j})_1, \dots, (\chi_{i,j})_{N_C}) \in W^h$, where $(\chi_{i,j})_k(q_l^k) := \delta_{ik}\delta_{jl}$ for $i,k = 1, \dots, N_C$, $j = j_0^i, \dots, N_i$, $l = j_0^k, \dots, N_k$, where $j_0^i = 1$ if Γ_i is closed and $j_0^i = 0$ else. We call $\{\chi_{i,j}\}_{i=1,\dots,N_C, j=j_0^i,\dots,N_i}$ standard basis of W^h.

Further, let $0 = t_0 < t_1 < \dots < t_M = T$ be a partitioning of the time interval $[0,T]$ into possibly variable time steps $\tau_m := t_{m+1} - t_m$, $m = 0, \dots, M-1$. Let $\vec{X}^m = (\vec{X}_1^m, \dots, \vec{X}_{N_C}^m) \in \underline{V}^h$ be an approximation of $\vec{x}(\,.\,, t_m) = (\vec{x}_1(\,.\,, t_m), \dots, \vec{x}_{N_C}(\,.\,, t_m))$, and let $\kappa^m = (\kappa_1^m, \dots, \kappa_{N_C}^m) \in W^h$ be an approximation of $\kappa(\,.\,, t_m) = (\kappa_1(\,.\,, t_m), \dots, \kappa_{N_C}(\,.\,, t_m))$.

Further, we make use of the shorthand notation $\vec{X}_{i,j}^m := \vec{X}_i^m(q_j^i)$ for $\vec{X}^m \in \underline{V}^h$ and $\kappa_{i,j}^m := \kappa_i^m(q_j^i)$ for $\kappa^m \in W^h$.

Let Γ_i^m be the image of $\vec{X}_i^m$, $i = 1, \dots, N_C$, and $\Gamma^m = (\Gamma_1^m, \dots, \Gamma_{N_C}^m)$. We introduce the finite element space

$$\underline{V}_\partial^h(\vec{X}^m) := \Big\{ (\vec{\eta}_1, \dots, \vec{\eta}_{N_C}) \in \underline{V}^h \,:\, \vec{\eta}_{i_{I,k}}(\rho_{I,k}) \,.\, \vec{n}_{\partial\Omega}(\vec{X}_{i_{I,k}}^m(\rho_{I,k})) = 0,\ \forall k = 1, \dots, N_I \Big\}. \tag{3.76c}$$

In the case of triple junctions let $j_{k,l} \in \{0, N_{i_{k,l}}\}$ denote the index of the corresponding curve endpoint such that $q^{i_{k,l}}_{j_{k,l}} = \rho_{k,l}$, $k = 1, \dots, N_T$, $l = 1, 2, 3$. In the case of boundary intersections let $j_{I,k} \in \{0, N_{i_{I,k}}\}$ denote the index of the corresponding curve endpoint such that $q^{i_{I,k}}_{j_{I,k}} = \rho_{I,k}$, for $k = 1, \dots, N_I$.

For scalar and vector functions $u = (u_1, \dots, u_{N_C}), v = (v_1, \dots, v_{N_C}) \in L^2(I_1, \mathbb{R}^{(2)}) \times \dots \times L^2(I_{N_C}, \mathbb{R}^{(2)})$, the L^2-inner product $\langle \cdot\,, \cdot \rangle_m$ over the current polygonal curve network Γ^m is given by

$$\langle u, v \rangle_m := \int_{\Gamma^m} u \,.\, v \,\mathrm{d}s := \sum_{i=1}^{N_C} \int_{I_i} u_i \,.\, v_i \, \|(\vec{X}^m_i)_\rho\| \,\mathrm{d}\rho. \tag{3.77}$$

Further, we make use of a mass lumped inner product as in Barrett et al. (2007a): The mass lumped inner product $\langle \,.\,, \,.\, \rangle^h_m$ for piecewise continuous functions $u = (u_1, \dots, u_{N_C})$ and $v = (v_1, \dots, v_{N_C})$ is defined by

$$\langle u, v \rangle^h_m := \frac{1}{2} \sum_{i=1}^{N_C} \sum_{j=1}^{N_i} h^m_{i,j-\frac{1}{2}} \left[(u_i \,.\, v_i) \left([q^i_j]^- \right) + (u_i \,.\, v_i) \left([q^i_{j-1}]^+ \right) \right], \tag{3.78}$$

where $g([q^i_j]^\pm) := \lim_{\epsilon \to 0, \epsilon > 0} g(q^i_j \pm \epsilon)$ for a function $g : I_i \to \mathbb{R}^{(2)}$, and $h^m_{i,j-\frac{1}{2}} := \|\vec{X}^m_{i,j} - \vec{X}^m_{i,j-1}\| > 0$ is the distance between the two neighbor nodes. Let

$$h^m := \max_{i=1,\dots,N_C, j=1,\dots,N_i} h^m_{i,j-\frac{1}{2}}$$

denote the maximum distance between two neighbor nodes of the polygonal curves.

Discrete Normal Vector Fields

Let $\vec{X}^m \in \underline{V}^h$ be a given parameterization of the polygonal curve network Γ^m satisfying the following assumption:

($\mathcal{A}$) The distance between two neighbor nodes of Γ^m is positive, i.e. $h^m_{i,j-\frac{1}{2}} > 0$ for $i = 1, \dots, N_C$, $j = 1, \dots, N_i$ and $\vec{X}^m(q^i_{j+1}) \neq \vec{X}^m(q^i_{j-1})$ for $i = 1, \dots, N_C$ and $j = 1, \dots, N_i$ if $\partial \Gamma^m_i = \emptyset$ and $j = 1, \dots, N_i - 1$ if $\partial \Gamma^m_i \neq \emptyset$.

This mild assumption is necessary such that the following discrete vector fields are well-defined or not equal to $\vec{0}$, respectively:

Let $\vec{\nu}^m := (\vec{\nu}^m_1, \dots, \vec{\nu}^m_{N_C})$ such that $\vec{\nu}^m_i$, given by

$$(\vec{\nu}^m_i)_{|[q^i_{j-1}, q^i_j]} := \vec{\nu}^m_{i,j-\frac{1}{2}} := \frac{\left(\vec{X}^m_{i,j} - \vec{X}^m_{i,j-1} \right)^\perp}{h^m_{i,j-\frac{1}{2}}},$$

is a discrete normal vector field on Γ^m_i, $i = 1, \dots, N_C$. For later use, we define the weighted approximating normal vector at $\vec{X}^m_{i,j}$ by

$$\vec{\omega}^m_{i,j} := \frac{h^m_{i,j-\frac{1}{2}} \vec{\nu}^m_{i,j-\frac{1}{2}} + h^m_{i,j+\frac{1}{2}} \vec{\nu}^m_{i,j+\frac{1}{2}}}{h^m_{i,j-\frac{1}{2}} + h^m_{i,j+\frac{1}{2}}} = \frac{\left(\vec{X}^m_{i,j+1} - \vec{X}^m_{i,j-1} \right)^\perp}{h^m_{i,j-\frac{1}{2}} + h^m_{i,j+\frac{1}{2}}}, \tag{3.79}$$

for $j = 1, \dots, N_i$ if $\partial\Gamma_i^m = \emptyset$ and for $j = 1, \dots, N_i - 1$ if $\partial\Gamma_i^m \neq \emptyset$. In the latter case, we set

$$\vec{\omega}_{i,0}^m := \vec{\nu}_{i,\frac{1}{2}}^m = \frac{(\vec{X}_{i,1}^m - \vec{X}_{i,0}^m)^\perp}{h_{i,\frac{1}{2}}^m}, \qquad \vec{\omega}_{i,N_i}^m := \vec{\nu}_{i,N_i-\frac{1}{2}}^m = \frac{(\vec{X}_{i,N_i}^m - \vec{X}_{i,N_i-1}^m)^\perp}{h_{i,N_i-\frac{1}{2}}^m}. \tag{3.80}$$

We set $\vec{\omega}^m := (\vec{\omega}_1^m, \dots, \vec{\omega}_{N_C}^m)$ with $\vec{\omega}_i^m(q_j^i) = \vec{\omega}_{i,j}^m$.

Discrete scheme

In the following, we always assume that the mild assumption $(\mathcal{A})$ holds for $\vec{X}^m$, $m = 0, \dots, M-1$.

We propose the following discrete scheme: Let $\vec{X}^0 \in \underline{V}^h$ be a given parameterization of a polygonal curve network Γ^0 which satisfies the attachment condition at triple junctions, in detail $\vec{X}_{i_{k,1}}^0(q_{j_{k,1}}^{i_{k,1}}) = \vec{X}_{i_{k,2}}^0(q_{j_{k,2}}^{i_{k,2}}) = \vec{X}_{i_{k,3}}^0(q_{j_{k,3}}^{i_{k,3}})$ for $k = 1, \dots, N_T$. For $m = 0, \dots, M-1$, find $\delta\vec{X}^{m+1} \in \underline{V}_{\partial}^h(\vec{X}^m)$, $\kappa^{m+1} \in W^h$ such that

$$\langle \frac{\delta\vec{X}^{m+1}}{\tau_m}, \chi\, \vec{\nu}^m \rangle_m^h - \sigma \langle \kappa^{m+1}, \chi \rangle_m^h = \langle F^m, \chi \rangle_m^h, \qquad \forall \chi \in W^h, \tag{3.81a}$$

$$\langle \kappa^{m+1}\, \vec{\nu}^m, \vec{\eta} \rangle_m^h + \langle \nabla_s \vec{X}^{m+1}, \nabla_s \vec{\eta} \rangle_m = 0, \qquad \forall \vec{\eta} \in \underline{V}_{\partial}^h(\vec{X}^m), \tag{3.81b}$$

is satisfied with $\vec{X}^{m+1} := \delta\vec{X}^{m+1} + \vec{X}^m$ and $F^m = (F_1^m, \dots, F_{N_C}^m) \in W^h$. Here, F_i^m, $i = 1, \dots, N_C$, is the piecewise linear function uniquely given by $F_i^m(q_j^i) := F_i(\vec{X}_{i,j}^m)$ for $j = j_0^i, \dots, N_i$. The function $F_i : \Gamma_i \to \mathbb{R}$, $i = 1, \dots, N_C$, is the external forcing term defined in (3.34) or the forcing term defined in (3.46b), (3.54b), (3.57b) in the case of color images. Coefficients like $c_k(t_m)$, for $k = 1, \dots, N_R$, are replaced by discrete counterparts c_k^m. We will later state how the coefficients c_k^m are computed in practice.

Note, that the finite element scheme (3.81) is the discrete analogue of (3.39).

Lemma 3.3. *Let $\delta\vec{X}^{m+1} = \vec{X}^{m+1} - \vec{X}^m \in \underline{V}_{\partial}(\vec{X}^m)$ and $\kappa^{m+1} \in W^h$ be a solution of the scheme* (3.81)*. Then, the attachment conditions at triple junctions* (3.36a) *and at boundary intersection points* (3.37a) *hold and for $h^m \to 0$, Young's law* (3.36b) *and the angle condition at boundary intersection points* (3.37b) *are satisfied.*

Proof. The attachment conditions are satisfied by $\delta\vec{X}^{m+1} \in \underline{V}_{\partial}(\vec{X}^m)$, the definition of $\vec{X}^{m+1}$ and the precondition $\vec{X}_{i_{k,1}}^0(q_{j_{k,1}}^{i_{k,1}}) = \vec{X}_{i_{k,2}}^0(q_{j_{k,2}}^{i_{k,2}}) = \vec{X}_{i_{k,3}}^0(q_{j_{k,3}}^{i_{k,3}})$ for $k = 1, \dots, N_T$.

Consider a triple junction $\vec{\Lambda}_k \in \Omega$, $k \in \{1, \dots, N_T\}$. Choose $\vec{\eta} = (\vec{\eta}_1, \dots, \vec{\eta}_{N_C}) \in \underline{V}_{\partial}^h(\vec{X}^m)$ with $\vec{\eta}_i(q_j^i) = \vec{e}_\alpha$ for $i = i_{k,l}$, $j = j_{k,l}$, $l = 1, 2, 3$, and $\vec{\eta}_i(q_j^i) = \vec{0}$ else, where $\vec{e}_\alpha$, $\alpha = 1, 2$, are the standard basic vectors in $\mathbb{R}^2$. For ease of illustration, we assume $j_{k,l} = 0$ for $l = 1, 2, 3$, i.e. the starting points of the three curves meet at the triple point. Testing (3.81b) with $\vec{\eta}$ leads to

$$0 = \sum_{l=1}^{3} \left(\frac{1}{2} h_{i_{k,l},\frac{1}{2}}^m \kappa_{i_{k,l}}^{m+1}(q_0^{i_{k,l}})\, \vec{\nu}_{i_{k,l},\frac{1}{2}}^m - \frac{\vec{X}_{i_{k,l}}^{m+1}(q_1^{i_{k,l}}) - \vec{X}_{i_{k,l}}^{m+1}(q_0^{i_{k,l}})}{h_{i_{k,l},\frac{1}{2}}^m} \right).$$

Considering $h^m \to 0$ results in Young's law

$$0 = \sum_{l=1}^{3} \vec{\tau}_{i_{k,l}}^{m+1}(\vec{\Lambda}_k),$$

as the first term is of order 1 w.r.t. h^m and $\left(\vec{X}^{m+1}_{i_{k,l}}(q_1^{i_{k,l}}) - \vec{X}^{m+1}_{i_{k,l}}(q_0^{i_{k,l}})\right)/h^m_{i_{k,l},\frac{1}{2}}$ approximates the tangent vector $\vec{\tau}^{m+1}_{i_{k,l}}(\vec{\Lambda}_k)$ at $\vec{\Lambda}_k$.

Consider a boundary intersection point $\vec{Q}_k \in \partial\Omega$, $k \in \{1,\dots,N_I\}$. Choose $\vec{\eta} = (\vec{\eta}_1,\dots,\vec{\eta}_{N_C}) \in \underline{V}^h_\partial(\vec{X}^m)$ with $\vec{\eta}_i(q^i_j) = \vec{n}_{\partial\Omega}(\vec{Q}_k)^\perp$ for $i = i_{I,k}$ and $j = j_{I,k}$ and $\vec{\eta}_i(q^i_j) = \vec{0}$ else. Again for ease of illustration, we assume $j_{I,k} = 0$, i.e. we assume that the boundary intersection point is the starting point of the curve. Testing (3.81b) with $\vec{\eta}$ leads to

$$0 = \frac{1}{2} h^m_{i_{I,k},\frac{1}{2}} \kappa^{m+1}_{i_{I,k}}(q_0^{i_{I,k}})\, \vec{\nu}^m_{i_{I,k},\frac{1}{2}} \,.\, \vec{n}_{\partial\Omega}(\vec{Q}_k)^\perp - \frac{\vec{X}^{m+1}_{i_{I,k}}(q_1^{i_{I,k}}) - \vec{X}^{m+1}_{i_{I,k}}(q_0^{i_{I,k}})}{h^m_{i_{I,k},\frac{1}{2}}} \,.\, \vec{n}_{\partial\Omega}(\vec{Q}_k)^\perp.$$

Considering $h^m \to 0$ results in the 90 degrees angle condition

$$0 = \vec{\tau}^{m+1}_{i_{I,k}}(\vec{Q}_k) \,.\, \vec{n}_{\partial\Omega}(\vec{Q}_k)^\perp,$$

since the first term is of order 1 w.r.t. h^m and the fraction in the second term approximates the tangent vector of the curve at $\vec{Q}_k$. □

Before we proceed to prove existence and uniqueness of a solution of the scheme (3.81), we make the following very mild assumptions, see also Barrett et al. (2007a,b).

($\mathcal{A}_1$) Let $i \in \{1,\dots,N_C\}$. If $\partial\Gamma^m_i = \emptyset$, we assume that $\dim \operatorname{span}\{\vec{\omega}^m_{i,j}\}_{j=1}^{N_i} = 2$.

($\mathcal{A}_2$) For each $k \in \{1,\dots,N_T\}$, we assume that $\dim \operatorname{span}\{\{\vec{\omega}^m_{i_{k,l},j}\}_{j=1}^{N_{i_{k,l}}-1}\}_{l=1}^{3} = 2$.

($\mathcal{A}_3$) For each $k \in \{1,\dots,N_I\}$, let $j^*_{I,k} = 1$ if $j_{I,k} = 0$ and $j^*_{I,k} = N_{i_{I,k}} - 1$ if $j_{I,k} = N_{i_{I,k}}$. We assume $\vec{X}^m_{i_{I,k}}(q^{i_{I,k}}_{j^*_{I,k}}) \notin \partial\Omega$.

Remark 3.4. *From assumption ($\mathcal{A}_3$), we obtain* $\dim \operatorname{span}\{\vec{\nu}^m_{i_{I,k},j_{I,k}\pm\frac{1}{2}}, \vec{n}_{\partial\Omega}(\vec{Q}_k)\} = 2$, *where the sign + is chosen in the case $j_{I,k} = 0$ and − is chosen in the case $j_{I,k} = N_{i_{I,k}}$.*

The assumptions are only violated in rare cases, cf. Barrett et al. (2007b): If a closed curve has no self-intersections, then ($\mathcal{A}_1$) always holds. A sufficient condition for ($\mathcal{A}_2$) to hold is that at least one of the three curves meeting at a triple junction is not a "saw tooth" like curve. Assumption ($\mathcal{A}_3$) holds if the contact angle between the polygonal curve and the boundary of the image domain $\partial\Omega$ is non-zero.

Theorem 3.5. *Let the assumptions ($\mathcal{A}$), ($\mathcal{A}_1$)-($\mathcal{A}_3$) hold. Then there exists a unique solution $(\delta\vec{X}^{m+1}, \kappa^{m+1}) \in \underline{V}^h_\partial(\vec{X}^m) \times W^h$ to the system (3.81).*

Proof. The system (3.81) is linear. Therefore, existence of a solution follows from its uniqueness. To prove uniqueness, consider the following system: Find $\{\vec{X}, \kappa\} \in \underline{V}^h_\partial(\vec{X}^m) \times W^h$ such that

$$\langle \vec{X}, \chi\, \vec{\nu}^m\rangle^h_m - \sigma\tau_m \langle \kappa, \chi\rangle^h_m = 0, \qquad \forall \chi \in W^h, \tag{3.82a}$$

$$\langle \kappa\, \vec{\nu}^m, \vec{\eta}\rangle^h_m + \langle \nabla_s \vec{X}, \nabla_s \vec{\eta}\rangle_m = 0, \qquad \forall \vec{\eta} \in \underline{V}^h_\partial(\vec{X}^m). \tag{3.82b}$$

We obtain choosing $\chi = \kappa \in W^h$ in (3.82a) and $\vec{\eta} = \vec{X} \in \underline{V}^h_\partial(\vec{X}^m)$ in (3.82b)

$$\sigma\tau_m \langle \kappa, \kappa\rangle^h_m + \langle \nabla_s \vec{X}, \nabla_s \vec{X}\rangle_m = 0.$$

From this equation, we conclude $\kappa \equiv 0$ and $\vec{X} \equiv \vec{X}^c$ for a constant $\vec{X}^c = (\vec{X}^c_1, \dots, \vec{X}^c_{N_C}) \in (\mathbb{R}^2)^{N_C}$ with $\vec{X}^c_{i_{k,1}} = \vec{X}^c_{i_{k,2}} = \vec{X}^c_{i_{k,3}}$ for all $k \in \{1, \dots, N_T\}$ and $\vec{X}^c_{i_{I,k}} \,.\, \vec{n}_{\partial\Omega}(\vec{Q}_k) = 0$ for all $k \in \{1, \dots, N_I\}$. Consequently, (3.82a) reduces to

$$\langle \vec{X}^c, \chi\, \vec{\nu}^m \rangle^h_m = 0, \quad \forall \chi \in W^h. \tag{3.83}$$

We now choose $\chi = \chi_{i,j} \in W^h$, $i \in \{1, \dots, N_C\}$, $j \in \{j^i_0, \dots, N_i\}$. Using the definitions of $\langle .\,, .\rangle^h_m$ and $\vec{\omega}^m_{i,j}$ yields

$$\vec{X}^c_i \,.\, \vec{\omega}^m_{i,j} = 0. \tag{3.84}$$

For closed curves Γ^m_i, $\vec{X}^c_i = \vec{0}$ follows from (3.84) and assumption $(\mathcal{A}_1)$. For an open curve Γ^m_i, the boundary nodes belong to either a triple junction or to a boundary intersection point in our setup.

Consider the case that one of the boundary nodes of Γ^m_i belong to a triple junction $\vec{\Lambda}_k$, $k \in \{1, \dots, N_T\}$, i.e. $i = i_{k,l}$ for $l \in \{1, 2, 3\}$. From $\vec{X}^c_{i_{k,1}} = \vec{X}^c_{i_{k,2}} = \vec{X}^c_{i_{k,3}}$ and (3.84), we get

$$\vec{X}^c_{i_{k,1}} \,.\, \vec{\omega}^m_{i_{k,l},j} = 0 \quad \text{ for all } l = 1, 2, 3,\ j = 1, \dots, N_{i_{k,l}} - 1. \tag{3.85}$$

With assumption $(\mathcal{A}_2)$, it follows that $\vec{X}^c_{i_{k,1}} = \vec{0}$ and thus $\vec{X}^c_i = \vec{0}$.

Let $\partial\Gamma^m_i$ intersect with the outer boundary $\partial\Omega$ at $\vec{Q}_k$, $k \in \{1, \dots, N_I\}$. Let $j = j_{I,k} \in \{0, N_i\}$. Without loss of generality, assume $j_{I,k} = N_i$. Choosing $\chi = \chi_{i,N_i}$ in (3.83) yields that $\vec{X}^c_i \,.\, \vec{\nu}^m_{i,N_i - \frac{1}{2}} = 0$. Since $\vec{X} \in \underline{V}^h_{\partial}(\vec{X}^m)$, we have $\vec{X}^c_i \,.\, \vec{n}_{\partial\Omega}(\vec{Q}_k) = 0$, and using assumption $(\mathcal{A}_3)$, we can conclude $\vec{X}^c_i = \vec{0}$. □

3.3.2 Solution of the Discrete System

A function $\phi = (\phi_1, \dots, \phi_{N_C}) \in W^h$ is piecewise linear and is therefore uniquely given by $\phi_{i,j} := \phi_i(q^i_j)$, $i = 1, \dots, N_C$, $j = j^i_0, \dots, N_i$, where $j^i_0 = 1$ if Γ^m_i is closed and $j^i_0 = 0$ else. Using the standard basis of W^h, we can write $\phi = \sum_{i=1}^{N_C} \sum_{j=j^i_0}^{N_i} \phi_{i,j} \chi_{i,j}$. In the following, we make use of a slight abuse of notation and consider $\phi = (\phi_1, \dots, \phi_{N_C})$ as an element of $\mathbb{R}^N$, with $N = \sum_{i=1}^{N_C} N^*_i$, where $N^*_i = N_i$ for closed curves and $N^*_i = N_i + 1$ for open curves. For $i = 1, \dots, N_C$ we write $\phi_i = (\phi_{i,j^i_0}, \dots, \phi_{i,N_i}) \in \mathbb{R}^{N^*_i}$. Thus, we consider a function in W^h as a vector of its coefficients with respect to the standard basis of W^h. Similarly, a function $\vec{\phi} = (\vec{\phi}_1, \dots, \vec{\phi}_{N_C}) \in \underline{V}^h$ can be interpreted as a vector of its coefficients, i.e. as a vector in $(\mathbb{R}^2)^N$, in detail $\vec{\phi}_i = (\vec{\phi}_{i,j^i_0}, \dots, \vec{\phi}_{i,N_i}) \in (\mathbb{R}^2)^{N^*_i}$.

Consequently, we consider $\kappa^{m+1} \in W^h$ and $\vec{X}^m, \vec{X}^{m+1} \in \underline{V}^h$ as elements in $\mathbb{R}^N$ and $(\mathbb{R}^2)^N$. Furthermore, we introduce an Euclidean space associated with $\underline{V}^h_{\partial}(\vec{X}^m)$:

$$\begin{aligned}\mathbb{X} := \Big\{ (\vec{z}_1, \dots, \vec{z}_{N_C}) \in (\mathbb{R}^2)^N \,:\, & [\vec{z}_{i_{k,1}}]_{j_{k,1}} = [\vec{z}_{i_{k,2}}]_{j_{k,2}} = [\vec{z}_{i_{k,3}}]_{j_{k,3}},\ k = 1, \dots N_T, \\ & \text{and } [\vec{z}_{i_{I,k}}]_{j_{I,k}} \,.\, \vec{n}_{\partial\Omega}(\vec{X}^m_{i_{I,k}, j_{I,k}}) = 0,\ k = 1, \dots, N_I \Big\},\end{aligned}$$

where $\vec{z}_i \in (\mathbb{R}^2)^{N^*_i}$ and $[\vec{z}_i]_j \in \mathbb{R}^2$ is the jth component of the vector $\vec{z}_i$. We consider $\delta\vec{X}^{m+1} = \vec{X}^{m+1} - \vec{X}^m \in \mathbb{X}$. For later use, let $\vec{P} : (\mathbb{R}^2)^N \to \mathbb{X}$ denote the orthogonal projection onto the space $\mathbb{X}$. Note, that $\mathbb{X}$ depends on $\vec{X}^m$. However, for simplicity, we do not use this dependency in the notation.

In order to state a matrix formulation for the discrete system (3.81), we introduce the following matrices:

$$M := \begin{pmatrix} M^1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & M^{N_C} \end{pmatrix}, \vec{N} := \begin{pmatrix} \vec{N}^1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \vec{N}^{N_C} \end{pmatrix}, \vec{A} := \begin{pmatrix} \vec{A}^1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \vec{A}^{N_C} \end{pmatrix},$$

where $M^i \in \mathbb{R}^{N_i^* \times N_i^*}$, $\vec{N}^i \in (\mathbb{R}^2)^{N_i^* \times N_i^*}$, $\vec{A}^i \in (\mathbb{R}^{2\times 2})^{N_i^* \times N_i^*}$, $i = 1, \dots, N_C$, are defined by

$$M^i_{jl} := \langle \chi_{i,j}, \chi_{i,l} \rangle^h_m, \quad \vec{N}^i_{jl} := \langle \chi_{i,j}, \chi_{i,l}\, \vec{\nu}^m \rangle^h_m, \quad \vec{A}^i_{jl} := \langle \nabla_s \chi_{i,j}, \nabla_s \chi_{i,l} \rangle_m \, \vec{\mathrm{Id}}_2,$$

where $\vec{\mathrm{Id}}_2 \in \mathbb{R}^{2\times 2}$ denotes the identity matrix.

Further, we define $b^m = (b^m_1, \dots, b^m_{N_C}) \in \mathbb{R}^N$ by

$$b^m_i = (b^m_{i,j^i_0}, \dots, b^m_{i,N_i}), \quad \text{with } \; b^m_{i,j} := \langle F^m_i, \chi_{i,j} \rangle^h_m, \; j = j^i_0, \dots, N_i, \tag{3.86}$$

for $i = 1, \dots, N_C$. Recall that $F^m = (F^m_1, \dots, F^m_{N_C})$, where $F^m_i := (F^m_{i,j^i_0}, \dots, F^m_{i,N_i}) \in \mathbb{R}^{N_i^*}$ and $F^m_{i,j} := F^m_i(q^i_j)$ is the external forcing term evaluated at the node $\vec{X}^m_{i,j}$.

We can rewrite the discrete system (3.81) to the following linear system: Find $\kappa^{m+1} \in \mathbb{R}^N$, $\delta X^{m+1} \in \mathbb{X}$ such that

$$\begin{pmatrix} -\sigma\tau_m M & \vec{N}^T \vec{P} \\ \vec{P}\vec{N} & \vec{P}\vec{A}\vec{P} \end{pmatrix} \begin{pmatrix} \kappa^{m+1} \\ \delta \vec{X}^{m+1} \end{pmatrix} = \begin{pmatrix} \tau_m b^m \\ -\vec{P}\vec{A}\vec{X}^m \end{pmatrix}, \tag{3.87}$$

The attachment condition at triple junctions and boundary intersection points are satisfied by $\delta X^{m+1} \in \mathbb{X}$ on assuming that $\vec{X}^0$ fulfills the attachment condition at triple junctions. Note that $\vec{P}^T = \vec{P}$ and $\vec{P}$ is the identity on $\mathbb{X}$.

As M is non-singular, (3.87) can be reformulated as

$$\kappa^{m+1} = \frac{1}{\sigma\tau_m} M^{-1} \left(\vec{N}^T \vec{P}\, \delta \vec{X}^{m+1} - \tau_m b^m \right), \tag{3.88a}$$

$$\left(\vec{P}\vec{A}\vec{P} + \frac{1}{\sigma\tau_m} \vec{P}\vec{N} M^{-1} \vec{N}^T \vec{P} \right) \delta \vec{X}^{m+1} = \frac{1}{\sigma} \vec{P}\vec{N} M^{-1} b^m - \vec{P}\vec{A}\vec{X}^m, \tag{3.88b}$$

by applying a Schur complement approach. The linear equation (3.88b) can be solved with an iterative solver, for example, with the method of conjugate gradients with possible preconditioning, or with a direct solver for sparse matrices.

Under the assumptions $(\mathcal{A}_1)$-$(\mathcal{A}_3)$ the system matrix in (3.88b) is symmetric and positive definite as a mapping $\mathbb{X} \to \mathbb{X}$. Hence, there exists a unique solution $\delta X^{m+1} \in \mathbb{X}$.

3.3.3 Equivalent Finite Difference Scheme

The linear system (3.87) can also be derived by following a finite difference approach. The finite element formulation (3.81) is based on the weak scheme (3.39). The finite difference scheme, introduced in this section, is a fully discrete scheme where the spatial and time derivatives in (3.35) are replaced by difference quotients.

As for the finite element scheme, let κ^{m+1}_i and $\vec{X}^{m+1}_i$ be piecewise linear and defined by their values at the nodes q^i_j, $i = 1, \dots, N_C$, $j = j^i_0, \dots, N_i$. Let $\vec{\omega}^{m+1}_{i,j}$ be a discrete

normal vector field defined as in (3.79) and (3.80). As in Section 3.3.2, we consider $\delta\vec{X}^{m+1} = \vec{X}^{m+1} - \vec{X}^m$ as an element in $(\mathbb{R}^2)^N$ and κ^{m+1} as an element in $\mathbb{R}^N$ with $N = \sum_{i=1}^{N_C} N_i^*$, $N_i^* = N_i$ for closed curves and $N_i^* = N_i + 1$ for open curves.

We propose the following approximation for the equation (3.35a):

$$\frac{1}{\tau_m}\delta\vec{X}_{i,j}^{m+1} \,.\, \vec{\omega}_{i,j}^{m} = \sigma\kappa_{i,j}^{m+1} + F_{i,j}^{m}, \tag{3.89}$$

for $i = 1, \ldots, N_C$, $j = j_0^i, \ldots, N_i$. Again, $F_{i,j}^m := F_i(\vec{X}_{i,j}^m)$ denotes the external forcing term evaluated at $\vec{X}_{i,j}^m$.

In order to propose a finite difference approximation of (3.35b), we need to define an approximation of $\vec{x}_{ss}(q_j^i, t_{m+1})$. For $i = 1, \ldots, N_C$ and $j = 1, \ldots, N_i$ if Γ_i^m is closed, and for $j = 1, \ldots, N_i - 1$ if Γ_i^m is not closed, we set

$$\Delta_2^{h,m}\vec{X}_{i,j}^{m+1} := \frac{2}{h_{i,j-\frac{1}{2}}^m + h_{i,j+\frac{1}{2}}^m}\left(\frac{\vec{X}_{i,j+1}^{m+1} - \vec{X}_{i,j}^{m+1}}{h_{i,j+\frac{1}{2}}^m} - \frac{\vec{X}_{i,j}^{m+1} - \vec{X}_{i,j-1}^{m+1}}{h_{i,j-\frac{1}{2}}^m}\right). \tag{3.90}$$

In the case of equal spatial step sizes $h_{i,j-\frac{1}{2}}^m = h_{i,j+\frac{1}{2}}^m =: h_i^m$, the term reduces to $(\vec{X}_{i,j-1}^m - 2\vec{X}_{i,j}^m + \vec{X}_{i,j+1}^m)/((h_i^m)^2)$.

An approximation of (3.35b) is given by

$$\kappa_{i,j}^{m+1}\vec{\omega}_{i,j}^{m} = \Delta_2^{h,m}\vec{X}_{i,j}^{m+1}, \tag{3.91}$$

for $i = 1, \ldots, N_C$, $j = 1, \ldots, N_i$ in the case of closed curves and for $j = 1, \ldots, N_i - 1$ in the case of open curves. Further, we request $\delta\vec{X}^{m+1}$ to lie in $\mathbb{X}$, such that the attachment conditions at triple junctions and boundary intersection points are satisfied. Again, let $\vec{P} : (\mathbb{R}^2)^N \to \mathbb{X}$ denote the orthogonal projection onto the space $\mathbb{X}$.

By recalling the definitions of M, $\vec{N}$, $\vec{A}$, b^m, $\langle\,.\,,\,.\,\rangle_m$ and $\langle\,.\,,\,.\,\rangle_m^h$, the linear scheme (3.87) includes (3.89) and (3.91). To show that, we multiply (3.89) and (3.91) with $\frac{1}{2}(h_{i,j-\frac{1}{2}}^m + h_{i,j+\frac{1}{2}}^m)$ for $i = 1, \ldots, N_C$ and for $j = 1, \ldots, N_i$ in the case of closed curves and for $j = 1, \ldots, N_i - 1$ in the case of open curves. For an open curve Γ_i^m, we multiply (3.89) with $\frac{1}{2}h_{i,\frac{1}{2}}^m$ for $j = 0$ and with $\frac{1}{2}h_{i,N_i-\frac{1}{2}}^m$ for $j = N_i$.

Young's law at triple junctions and the 90° angle condition at the image boundary can be shown for the discrete curves from the linear scheme (3.87) as in Lemma 3.3.

Vice versa, we can also obtain the linear system (3.87) from a finite difference approach by using (3.89) and (3.91) and the conditions at triple junctions and boundary intersection points. For details, we refer to Benninghoff and Garcke (2014), where the matrix entries of M, $\vec{N}$, $\vec{A}$ and the components of b^m are defined without using $\langle\,.\,,\,.\,\rangle_m$ and $\langle\,.\,,\,.\,\rangle_m^h$.

To sum up, the linear system (3.87) can be obtained not only by a finite element approximation but also by a finite difference approximation.

3.3.4 Semidiscrete Scheme

In this section, we consider a scheme which is discrete in space and continuous in time. Therefore, let $\vec{X} = (\vec{X}_1, \ldots, \vec{X}_{N_C})$ and $\kappa = (\kappa_1, \ldots, \kappa_{N_C})$ with $\vec{X}_i : I_i \times [0, T] \to \mathbb{R}^2$, and

$\kappa_i : I_i \times [0,T] \to \mathbb{R}$, $i = 1,\dots,N_C$, such that $\vec{X}(\,.\,,t)$ and $\kappa(\,.\,,t)$ are piecewise linear on $[q^i_{j-1}, q^i_j]$, $j = 1,\dots,N_i$, for each $t \in [0,T]$. We make use of a similar notation as in the fully discrete case by just omitting the superscripts m and $m+1$. All quantities are time-dependent in this section. To be precise, we set

$$\vec{X}_{i,j} := \vec{X}_i(q^i_j), \quad \kappa_{i,j} = \kappa_i(q^i_j), \quad h_{i,j-\frac{1}{2}} = \|\vec{X}_{i,j} - \vec{X}_{i,j-1}\|$$

and a normal vector field is given by

$$(\vec{\nu}_i)_{|[q^i_{j-1},q^i_j]} := \vec{\nu}_{i,j-\frac{1}{2}} := \frac{(\vec{X}_{i,j} - \vec{X}_{i,j-1})^\perp}{h_{i,j-\frac{1}{2}}}. \tag{3.92}$$

A node-wise weighted normal is defined by

$$\vec{\omega}_{i,j} := \frac{h_{i,j-\frac{1}{2}}\vec{\nu}_{i,j-\frac{1}{2}} + h_{i,j+\frac{1}{2}}\vec{\nu}_{i,j+\frac{1}{2}}}{h_{i,j-\frac{1}{2}} + h_{i,j+\frac{1}{2}}} = \frac{(\vec{X}_{i,j+1} - \vec{X}_{i,j-1})^\perp}{h_{i,j-\frac{1}{2}} + h_{i,j+\frac{1}{2}}}, \tag{3.93}$$

for $i = 1,\dots,N_C$ and $j = 1,\dots,N_i$ if $\partial\Gamma_i = \emptyset$ and for $j = 1,\dots,N_i - 1$ if $\partial\Gamma_i \neq \emptyset$. If $\partial\Gamma_i \neq \emptyset$, we set

$$\vec{\omega}_{i,0} := \vec{\nu}_{i,\frac{1}{2}} = \frac{(\vec{X}_{i,1} - \vec{X}_{i,0})^\perp}{h_{i,\frac{1}{2}}}, \quad \vec{\omega}_{i,N_i} := \vec{\nu}_{i,N_i-\frac{1}{2}} = \frac{(\vec{X}_{i,N_i} - \vec{X}_{i,N_i-1})^\perp}{h_{i,N_i-\frac{1}{2}}}. \tag{3.94}$$

We define the following difference quotient for non-boundary nodes

$$\Delta^h_2 \vec{X}_{i,j} := \frac{2}{h_{i,j-\frac{1}{2}} + h_{i,j+\frac{1}{2}}} \left(\frac{\vec{X}_{i,j+1} - \vec{X}_{i,j}}{h_{i,j+\frac{1}{2}}} - \frac{\vec{X}_{i,j} - \vec{X}_{i,j-1}}{h_{i,j-\frac{1}{2}}} \right). \tag{3.95}$$

We now state a result which demonstrates that the semidiscrete scheme leads to equidistributed meshes:

Lemma 3.6. *The semidiscrete scheme*

$$(\vec{X}_{i,j})_t \,.\, \vec{\omega}_{i,j} = \sigma\kappa_{i,j} + F_{i,j}, \tag{3.96a}$$

$$\kappa_{i,j}\,\vec{\omega}_{i,j} = \Delta^h_2 \vec{X}_{i,j}, \tag{3.96b}$$

for $i = 1,\dots,N_C$, $j = 1,\dots,N_i$ if Γ_i is open, and for $j = 1,\dots,N_i - 1$ if Γ_i is closed, provides an equidistribution of the mesh points along those parts of the curves Γ_i which are not locally flat.

Proof. (cf. also Barrett et al. (2007b)) We insert the definitions of $\vec{\omega}_{i,j}$ and $\Delta^h_2\vec{X}_{i,j}$ in (3.96b) and obtain

$$\kappa_{i,j} \frac{(\vec{X}_{i,j+1} - \vec{X}_{i,j-1})^\perp}{h_{i,j-\frac{1}{2}} + h_{i,j+\frac{1}{2}}} = \frac{2}{h_{i,j-\frac{1}{2}} + h_{i,j+\frac{1}{2}}} \left(\frac{\vec{X}_{i,j+1} - \vec{X}_{i,j}}{h_{i,j+\frac{1}{2}}} - \frac{\vec{X}_{i,j} - \vec{X}_{i,j-1}}{h_{i,j-\frac{1}{2}}} \right).$$

Taking the inner product with $\vec{X}_{i,j+1} - \vec{X}_{i,j-1}$ on both sides leads to

$$0 = \left(\frac{\vec{X}_{i,j+1} - \vec{X}_{i,j}}{h_{i,j+\frac{1}{2}}} - \frac{\vec{X}_{i,j} - \vec{X}_{i,j-1}}{h_{i,j-\frac{1}{2}}} \right) \cdot \left(\vec{X}_{i,j+1} - \vec{X}_{i,j-1} \right). \tag{3.97}$$

Writing $\vec{X}_{i,j+1} - \vec{X}_{i,j-1} = \vec{X}_{i,j+1} - \vec{X}_{i,j} + \vec{X}_{i,j} - \vec{X}_{i,j-1}$ implies

$$0 = h_{i,j+\frac{1}{2}} + \frac{\vec{X}_{i,j+1} - \vec{X}_{i,j}}{h_{i,j+\frac{1}{2}}} \cdot \left(\vec{X}_{i,j} - \vec{X}_{i,j-1}\right) - \frac{\vec{X}_{i,j} - \vec{X}_{i,j-1}}{h_{i,j-\frac{1}{2}}} \cdot \left(\vec{X}_{i,j+1} - \vec{X}_{i,j}\right) - h_{i,j-\frac{1}{2}}.$$

Multiplying with $h_{i,j+\frac{1}{2}} h_{i,j-\frac{1}{2}}$ and summarizing the firth with the forth and the second with the third term provides

$$0 = (h_{i,j-\frac{1}{2}} - h_{i,j+\frac{1}{2}}) \left[(\vec{X}_{i,j+1} - \vec{X}_{i,j}) \cdot (\vec{X}_{i,j} - \vec{X}_{i,j-1}) - h_{i,j+\frac{1}{2}} h_{i,j-\frac{1}{2}} \right].$$

This equation is true if either

$$h_{i,j+\frac{1}{2}} = h_{i,j-\frac{1}{2}}, \qquad \text{i.e. } \|\vec{X}_{i,j+1} - \vec{X}_{i,j}\| = \|\vec{X}_{i,j} - \vec{X}_{i,j-1}\| \tag{3.98}$$

or

$$(\vec{X}_{i,j+1} - \vec{X}_{i,j}) \parallel (\vec{X}_{i,j} - \vec{X}_{i,j-1}) \tag{3.99}$$

holds. Consequently, two neighbor line segments of the polygonal curve Γ_i have either the same length or the three node points $\vec{X}_{i,j-1}, \vec{X}_{i,j}, \vec{X}_{i,j+1}$ lie on one straight line. □

Remark 3.7. *The equidistribution of mesh points can also be derived by considering a continuous in time, semidiscrete finite element scheme, cf. Barrett et al. (2007b):*

$$\langle \vec{X}_t, \chi\, \vec{\nu} \rangle^h - \sigma \langle \kappa, \chi \rangle^h = \langle F, \chi \rangle^h, \qquad \forall \chi \in W^h, \tag{3.100a}$$

$$\langle \kappa\, \vec{\nu}, \vec{\eta} \rangle^h + \langle \nabla_s \vec{X}, \nabla_s \vec{\eta} \rangle = 0, \qquad \forall \vec{\eta} \in \underline{V}^h_\partial, \tag{3.100b}$$

where

$$\underline{V}^h_\partial := \left\{ (\vec{\eta}_1, \ldots, \vec{\eta}_{N_C}) \in \underline{V}^h \,:\, \vec{\eta}_{i_{I,k}}(\rho_{I,k}) \cdot \vec{n}_{\partial\Omega}(\vec{X}_{i_{I,k}}(\rho_{I,k})) = 0,\ \forall k = 1, \ldots, N_I \right\}. \tag{3.101}$$

and the product $\langle \,.\,, \,.\, \rangle^{(h)}$ *is defined as in* (3.77) *and* (3.78) *by replacing* $\vec{X}^m$ *by* $\vec{X}(\,.\,,t)$ *and* Γ^m_i *by the current polygonal curve* $\Gamma_i(t)$, $i = 1, \ldots, N_C$, $t \in [0,T]$.

Testing (3.100b) *with* $\vec{\eta} = \chi_{i,j} (\vec{\omega}_{i,j})^\perp$, $i = 1, \ldots, N_C$, $j = 1, \ldots, N_i$ *if* $\partial\Gamma_i = \emptyset$ *and* $j = 1, \ldots, N_i - 1$ *else, leads to* (3.97).

For the fully discrete scheme we cannot prove the equidistribution of mesh points. However, practical experiments show that the nodes are well distributed, and therefore no *local* refinement or coarsening strategy need be developed. As a curve can grow and shrink during the evolution, a *global* refinement or coarsening is recommended, see Section 3.3.6.

3.3.5 Topology Changes

During the evolution of curves the following topology changes can occur:

- splitting of a curve into two curves,
- merging of two curves into a single curve,
- emergence of a triple junction,
- emergence of a boundary intersection point, and
- deletion of a curve.

Also two or more topology changes can occur at the same time like, for example, splitting of a curve at different locations.

If triple junctions are detected after a touch of two curves, a new curve will be created. Similarly, if a curve length shrinks, the curve has to be deleted. If the curve connects two triple junctions, the triple junctions will also be deleted, and other curves which previously ended at the junctions have to be adapted. Therefore, the number of curves N_C^m, the number of nodes N_i^m, $i = 1, \dots N_C^m$, the number of triple junctions N_T^m, and the number of boundary intersection points N_I^m are time-dependent.

Using a parametric approach, topology changes are not automatically handled, in contrast to level set methods for example. Using the level set method, some topology changes like splitting, merging, deletion of curves and intersection with the image boundary are handled automatically without the need of an additional routine. With a parametric approach, topology changes can be interpreted as singularities, cf. Deckelnick et al. (2005). We propose a method for continuing after such singularities occur. The detection of the topology changes used in this work is close to the method proposed by Balažovjech et al. (2012); Mikula and Urbán (2012), who consider a two-phase setup with one interfacing curve without triple junctions and boundary intersections. We generalize their approach such that multiple phases with junctions and boundary points also can be dealt with.

Figure 3.8 shows a diagram of the algorithm for detecting topology changes. To detect a change in topology, we construct a uniform background grid which covers the rectangular image domain Ω. The grid size $a > 0$ can be adaptively chosen; for example, it can be set according to the minimum or average distance between two neighbor node points of the polygonal curves. Alternatively, a can be chosen dependent on the maximum norm of the position deviation $\max\{\|\delta X_{i,j}^{m+1}\|, i = 1, \dots, N_C, j = 0, \dots, N_i^m\}$. This choice of a ensures that the algorithm is robust with respect to the choice of the time step size. By using a large grid size a, large time step sizes also can be dealt with. The grid can be stored as a sparse matrix or two-dimensional array, where the elements correspond to squares of the grid of size $a \times a$. Using a background grid, topology changes can efficiently be detected in two steps.

(i) For $i = 1, \dots, N_C^m$ and $j = 0, \dots, N_i^m$, we initialize the square in which $\vec{X}_{i,j}^m$ lies with $(-1, -1)$. If the square is close to the boundary $\partial\Omega$, the node is marked as a boundary intersection point. Therefore, it is stored in a list for boundary intersection points such that we can handle several topology changes at once.

(ii) In a second loop, we again consider for $i = 1, \dots, N_C^m$ and $j = 0, \dots, N_i^m$ the corresponding square of the grid. If the square is marked with $(-1, -1)$, the square marking is overwritten with (i, j). If the square has already been marked with (i_1, j_1), a topology change is likely to occur close to the nodes. If $i = i_1$ and if the two nodes are direct neighbor points or have a common neighbor, no topology change is detected. Otherwise, a set of a limited number n of neighbor nodes around each $\vec{X}_{i,j}^m$ and $\vec{X}_{i_1,j_1}^m$ is considered. Since each of the two sets contains only n nodes, the pair (i, j^*) and (i_1, j_1^*) with the smallest distance can be quickly found. In practice, $n = 5$ is a good choice. If $i = i_1$, a splitting of the curve Γ_i^m occurs and the pair (i, j^*) and (i_1, j_1^*) is stored in a list for splitting points. If $i \neq i_1$, we consider $k^+(i), k^-(i), k^+(i_1)$ and $k^-(i_1)$, i.e. the indices of the regions Ω_k, $k \in \{1, \dots, N_R\}$, separated by Γ_i^m and $\Gamma_{i_1}^m$, respectively. Recall that the normal $\vec{\nu}_i^m$ at the curve Γ_i^m points from $k^-(i)$ to $k^+(i)$. If $k^+(i) = k^+(i_1)$ and $k^-(i) = k^-(i_1)$, the two curves separate the same two regions and a merging of the two curves occurs. The pair is stored in a list for merging points. We exclude the case $k^+(i) = k^-(i_1)$ and $k^-(i) = k^+(i_1)$, as this situation can never occur

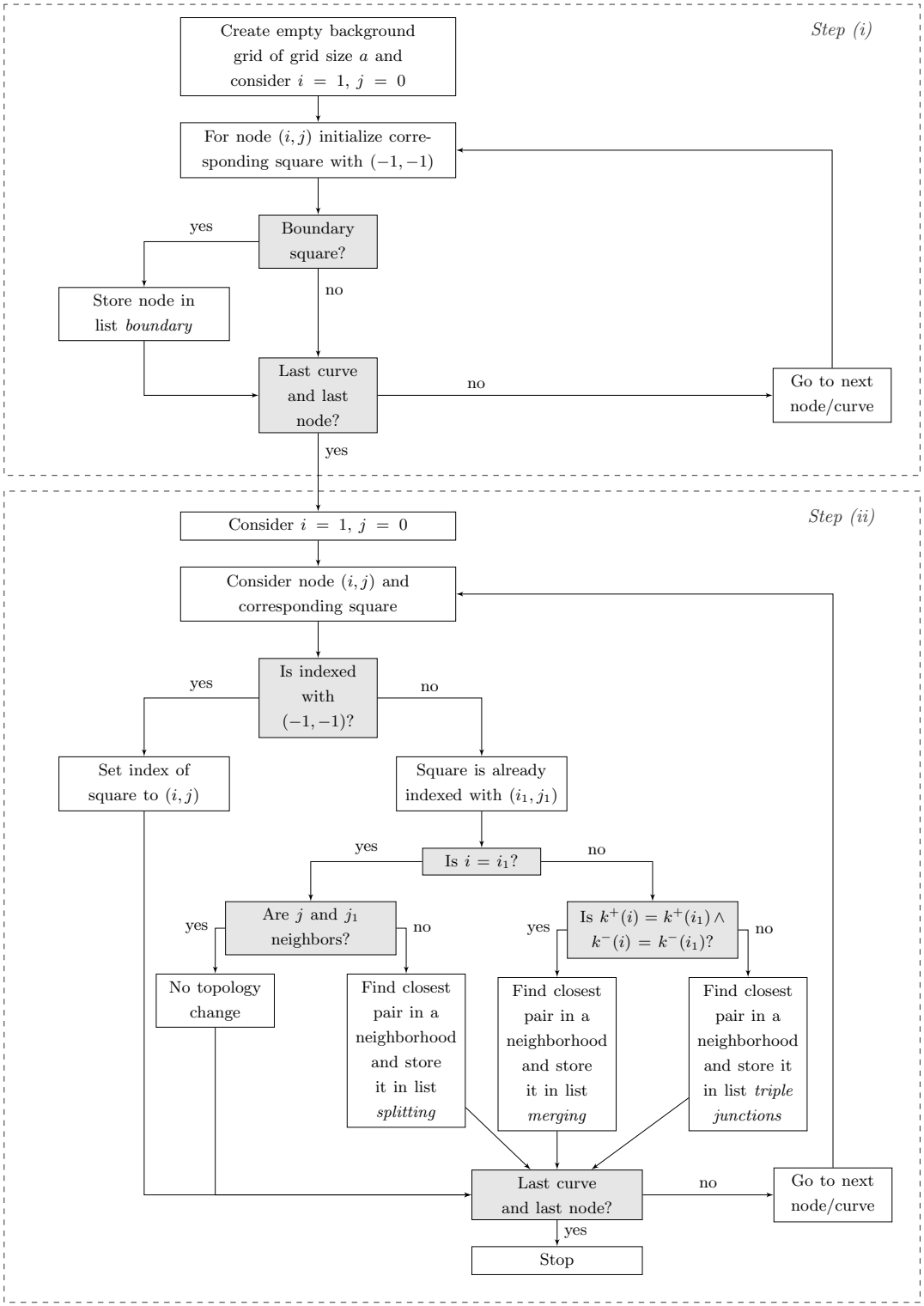


Figure 3.8: Illustration of the detection of topology changes.

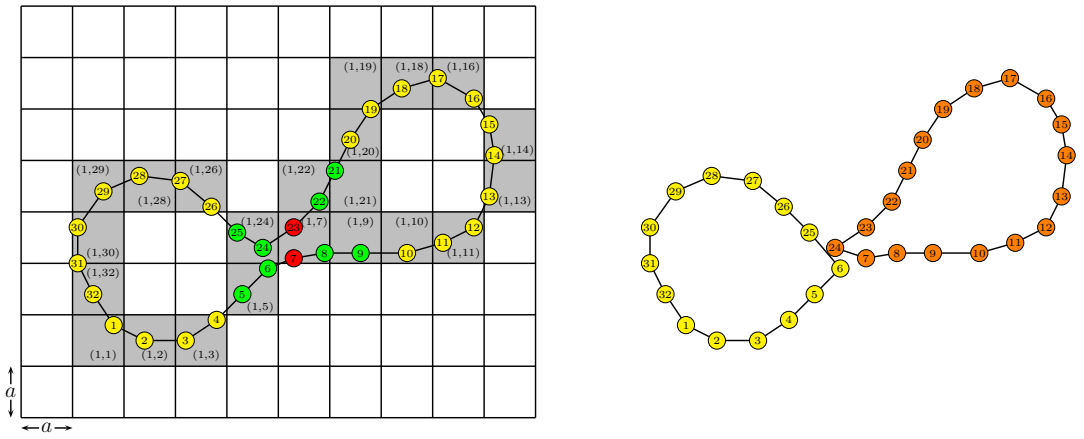


Figure 3.9: Left: Detection of splitting of a curve Γ_1^m near the nodes 7 and 23, cf. Balažovjech et al. (2012) (Fig. 2.2) and Mikula and Urbán (2012) (Fig. 2). Right: Splitting of the curve into two curves.

when taking care of the orientation of the initial curves. In the case of $k^+(i) \neq k^+(i_1)$ or $k^-(i) \neq k^- i_1)$, triple junctions occur and the pair of nodes is stored in a list for triple junctions.

Topology changes can thus be detected efficiently by performing only two loops over all mesh points. Consequently, the computational effort for detecting topology changes remains small; more precisely it is $\mathcal{O}(N)$. The algorithm can handle several topology changes at once. The lists of split, merge, triple, and boundary points can be sequentially considered. In the case of splitting and merging of curves, the neighbor relation is modified by connecting node j^* with $j_1^* + 1$ and node j_1^* with $j^* + 1$.

Figure 3.9 presents an example where a curve splits into two single curves: The node $(1, 23)$ lies in a square which has been previously marked with $(1, 7)$ (left subfigure). Among the mesh points in a neighborhood of $(1, 7)$ and in a neighborhood of $(1, 23)$, the pair $(1, 6)$ and $(1, 24)$ is identified as the pair of nodes with smallest distance. The neighbor relationships at $(1, 6)$ and $(1, 24)$ need to be modified as shown in Figure 3.9 (right). To prevent another splitting and merging near the same point in Ω, a square can be blocked for new topology changes for some time steps after a topology change has taken place.

If a boundary point occurs, the point is first projected orthogonally to the boundary $\partial\Omega$ and is then duplicated. If the previous curve is closed, the curve becomes an open curve. If the curve has already been an open curve, it is separated into two single curves. The two boundary nodes, the detected node and its duplication, are the end point of one curve and

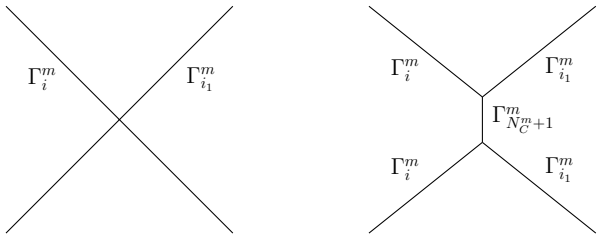


Figure 3.10: Conversion of a quadruple junction to a network with two triple junctions.

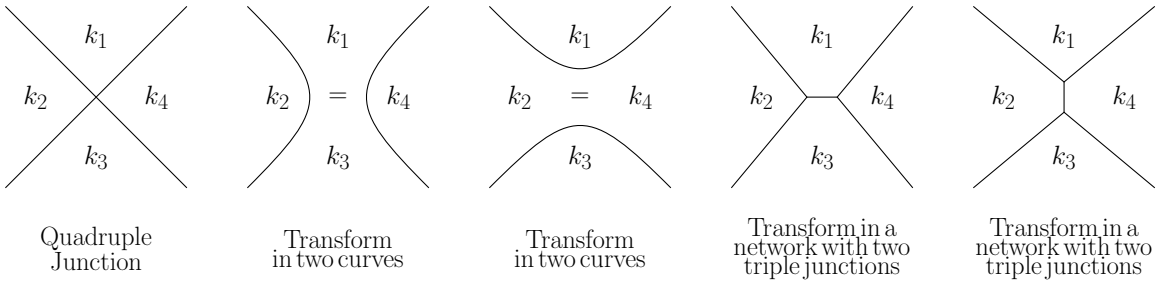


Figure 3.11: Quadruple junction and several possibilities for continuation.

the first point of the second curve. They are allowed to move away from each other along $\partial\Omega$ in the next time steps. Similarly, the complementary situation can be handled. An open curve can become a closed curve if the two boundary points are too close to each other, i.e. lie in the same square of the background grid. Two curves can merge if their boundary points are too close to each other. Again, the square where a boundary intersection has taken place can be blocked for a few time steps.

In the case of triple junctions, the topology change is handled as follows: As an intermediate step, a quadruple junction emerges at the point where Γ_i^m and $\Gamma_{i_1}^m$ touch. The quadruple junction is replaced by two triple junctions connected with a small curve, see Figure 3.10. The small curve consists of only two initial points, namely $0.5(\vec{X}_{i,j^*}^m + \vec{X}_{i_1,j_1^*}^m)$ and $0.5(\vec{X}_{i,j^*\pm1}^m + \vec{X}_{i_1,j_1^*\pm1}^m)$, where the sign $\pm$ is chosen according to the orientation of the curves. Within the next time steps, the new curve can grow and new nodes are inserted by some global refinements, cf. Section 3.3.6.

The algorithm can also handle complex cases with more than one topology change at the same time. Some exemplary cases are discussed in Appendix 3.A.

Another group of topology changes occur when a curve's length falls below a minimum length. If the curve is closed or if its two boundary nodes both lie on $\partial\Omega$, it is directly deleted. If both end nodes belong to triple junctions, the curve is deleted and an intermediate quadruple junction occurs. Dependent on the phase indices of the adjacent regions Ω_k, $k \in \{1, \ldots, N_R\}$, and of the previous curve network, there are different ways of continuing, see Figure 3.11. If a curve is marked for deletion with one boundary belonging to a triple junction and one boundary located at $\partial\Omega$, the curve and the triple junction are deleted. Further, the boundary points of the remaining two curves which have met at the triple junction are projected to the boundary $\partial\Omega$. In the next time steps, they are allowed to move away from each other.

3.3.6 Additional Computational Aspects

Computations of regions and coefficients

As the curves evolve in time, an approximation Ω_k^m of the regions $\Omega_k(t_m)$ and an approximation c_k^m of the coefficients $c_k(t_m)$ (cf. (3.23)), $k = 1, \ldots, N_R$, have to be computed. As the image function is locally constant at the pixels, we assign each pixel to a phase Ω_k^m. If a pixel is truncated by a curve, it is assigned to the phase to which the largest part belongs. In the rare case, that a pixel is truncated by a curve Γ_i^m in two parts of equal size, the pixel is assigned either to $\Omega_{k^+(i)}^m$ or $\Omega_{k^-(i)}^m$.

We first consider the case of a scalar, gray-scaled image with image function $u_0 : \Omega \to \mathbb{R}$. Let S_k^m be the set of n_k^m pixels belonging to Ω_k^m. Then the approximation c_k^m is set to

$$c_k^m := \frac{C_k^m}{n_k^m}, \qquad C_k^m := \sum_{pix \in S_k^m} u_0|_{pix}. \tag{3.102}$$

The entire image domain needs to be considered only for $m = 0$. As the curves move only slightly from one time step to the next, the regions Ω_k^m and the coefficients c_k^m need to be updated considering only a small environment of the curves. This results in a very efficient computation of the regions and the coefficients. As the normal $\vec{\nu}_i^m$ points from $\Omega_{k^-(i)}^m$ to $\Omega_{k^+(i)}^m$, the pixels close to the curve Γ_i^m are assigned to the phase $k^+(i)$ or $k^-(i)$, respectively.

As initialization we set $n_k^m = n_k^{m-1}$ and $C_k^m = C_k^{m-1}$ for $k = 1, \ldots, N_R$. For $i = 1, \ldots, N_C^m$, all pixels in an environment of Γ_i^m are subsequently considered. Let a pixel pix be assigned to phase $k \in \{k^+(i), k^-(i)\}$ and let $l \neq k$ be the former phase index of the pixel. We set

$$n_k^m = n_k^m + 1, \quad n_l^m = n_l^m - 1, \quad C_k^m = C_k^m + u_0|_{pix}, \quad C_l^m = C_l^m - u_0|_{pix}. \tag{3.103}$$

After having considered all pixels close to the curves, the coefficients are set to $c_k^m = C_k^m / n_k^m$ for $k = 1, \ldots, N_R$.

Similarly, the coefficients in the case of colored images can be computed. The coefficients are all means or normalized means of the corresponding component of the image function. In the case of CB or HSV color space, the value of a given RGB image $\vec{u_0}|_{pix}$ needs to be first transformed to the CB and HSV color. If image components lie on S^1 or S^2, the computation needs to be slightly modified: For example, in the case of the HSV space, the hue coefficients are computed by $\vec{h}_k^m = \vec{H}_k^m / \|\vec{H}_k^m\|$ with $\vec{H}_k^m = \sum_{pix \in S_k^m} \vec{h}_0|_{pix}$, where $\vec{h}_0 : \Omega \to S^2$ is the hue component of the image (cf. (3.56)).

Global refinement-coarsening strategy

We propose a simple global refinement and coarsening strategy. The lengths $|\Gamma_i^m|$ of the polygonal curves Γ_i^m, $i = 1, \ldots, N_C$, have been already computed since the length is used as criterion to delete a curve, recall Section 3.3.5. As in the notation above, let N_i^m be the (time-dependent) number of nodes belonging to the curve Γ_i^m. If the term $|\Gamma_i^m| / N_i^m$ is smaller than a threshold $l_{\min} > 0$, we perform a global coarsening, i.e. we delete every second node of the polygonal curve which is not an endpoint. If $|\Gamma_i^m| / N_i^m > l_{\max}$ for a threshold $l_{\max} > l_{\min}$, we perform a global refinement of the curve discretization by inserting a new node between two neighbor nodes.

Adaptive choice of weighting parameter σ

We can use a constant parameter σ which weights the curvature term in (3.35a). Alternatively, the parameter can be set adaptively: For this, the internal (or perimeter) energy $\sigma|\Gamma|$ in the energy functional (3.20) is set to a certain percentage of the external energy $E(\Gamma, c_1, \ldots, c_{N_R}) - \sigma|\Gamma|$. This percentage is called the σ-factor for short. The energies and the new value for σ need not be updated at each time step. In the results presented in Section 3.4, the factor σ is computed either every 10th or every 50th time step.

Setting $\sigma|\Gamma|$ to a certain percentage of the external energy provides robustness with respect to noise: If the noise is high, the external energy will be large, and the weight σ will be set

larger compared to small noise. Thus, in the case of high noise, a large value of σ provides sufficient smoothness of the curve. The percentage of the external energy (σ-factor) can be set according to the desired scale of features that should be detected. The σ-factor will be chosen smaller if smaller features should be resolved.

Time step control

The computed energy can also be used to implement a simple time step size control: Let $\Delta t = \tau_m$ denote the (possibly variable) time step size. Let $E_{\text{rel}}(m)$ denote the external energy at time step t_m divided by the external energy at t_0. We compare the current relative energy $E_{\text{rel}}(m)$ with the relative energy at time step $m - \Delta m$, where Δm is the number of time steps between two computations of the energy, e.g. $\Delta m = 10$. If the decrease $E_{\text{rel}}(m-\Delta m) - E_{\text{rel}}(m)$ is larger than a threshold $\Delta E_{\max}$, the time step size is decreased and set to $0.5\Delta t$. If the difference is smaller than a threshold $\Delta E_{\min}$ (with $\Delta E_{\min} < \Delta E_{\max}$), the time step size is increased and set to $2\Delta t$. Thus we can speed up the image segmentation when the energy decrease is getting too slow.

The energy decrease can also be used as a stopping criterion: If the energy difference is less than a threshold ΔE_{del}, with $\Delta E_{\text{del}} \ll \Delta E_{\max}$, the curve evolution is stopped and we can assume that the image segmentation is finished.

3.3.7 Numerical Solution of the Image Restoration Scheme

In addition to the segmentation of the image, the algorithm provides a piecewise constant approximation $u = \sum_{k=1}^{N_R} c_k^m \chi_{\Omega_k^m}$ of u_0. A piecewise smooth approximation can be obtained by solving a discretization of the boundary value problem (3.43). For the bulk equations, a finite difference approximation can be used. In this section, we describe in detail how the diffusion equation with Neumann boundary conditions is solved numerically.

We consider a spatial discretization

$$\Omega^h := \{(ih, jh) \,:\, i = 0, \ldots, N_x,\, j = 0 \ldots, N_y\}\,, \quad N_x, N_y \in \mathbb{N},$$

of the rectangular image domain Ω. As we have to solve an equation on each phase separately, we define the discrete set $\Omega_k^h := \Omega^h \cap \overline{\Omega_k^m}$. In the following, we consider k as fixed. We aim at finding a function u^h minimizing a discrete analogue of the energy (3.41).

We define for $i = 1, \ldots, N_x$, $j = 1, \ldots, N_y$

$$A_x(i,j) = area\left(\left([(i-1)h, ih] \times [(j-\frac{1}{2})h, (j+\frac{1}{2})h]\right) \cap \Omega_k^m\right),$$
$$A_y(i,j) = area\left(\left([(i-\frac{1}{2})h, (i+\frac{1}{2})h] \times [(j-1)h, jh]\right) \cap \Omega_k^m\right),$$

and for $i = 0, 1, \ldots, N_x$, $j = 0, 1, \ldots, N_y$

$$A(i,j) = area\left(([(i-\frac{1}{2})h, (i+\frac{1}{2})h] \times [(j-\frac{1}{2})h, (j+\frac{1}{2})h]) \cap \Omega_k^m\right),$$

where *area* denotes the two-dimensional Lebesgue measure.

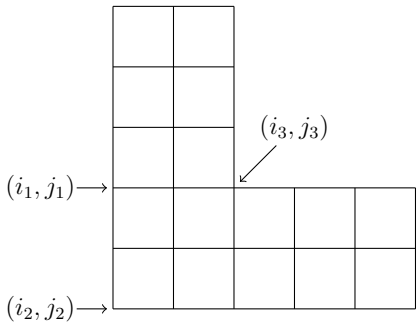


Figure 3.12: A possible region Ω_k^m.

We define the following discrete energy:

$$E_{\text{discr},k}(u^h) = \sum_{i=1}^{N_x}\sum_{j=1}^{N_y}\left(A_x(i,j)\left(\frac{u_{i,j}^h - u_{i-1,j}^h}{h}\right)^2 + A_y(i,j)\left(\frac{u_{i,j}^h - u_{i,j-1}^h}{h}\right)^2\right) + \sum_{i=0}^{N_x}\sum_{j=0}^{N_y}\lambda_k A(i,j)\left(u_{i,j}^h - u_0(ih,jh)\right)^2, \tag{3.104}$$

which is a discrete analogue of $\int_{\Omega_k}\left((u_x^2+u_y^2)+\lambda_k(u-u_0)^2\right)\mathrm{d}x$. For each $(ih,jh)\in\Omega_k^h$ we take the derivative with respect to $u_{i,j}^h$ and set the resulting term to zero. This leads to a linear system. Recall that Ω_k^m need not be rectangular. Geometries like the one shown in Figure 3.12 can occur.

At interior nodes (ih,jh), we get the following equation considering the derivative with respect to $u_{i,j}^h$:

$$\left(4+\lambda_k h^2\right)u_{i,j}^h - u_{i-1,j}^h - u_{i+1,j}^h - u_{i,j-1}^h - u_{i,j+1}^h = \lambda_k h^2 u_0(ih,jh).$$

At boundary points like $(i_1,j_1),(i_2,j_2),(i_3,j_3)$, we get the following equations from (3.104):

$$\left(2+\frac{1}{2}\lambda_k h^2\right)u_{i_1,j_1}^h - u_{i_1+1,j_1}^h - \frac{1}{2}u_{i_1,j_1-1}^h - \frac{1}{2}u_{i_1,j_1+1}^h = \frac{1}{2}\lambda_k h^2 u_0(i_1h,j_1h), \tag{3.105}$$

$$\left(1+\frac{1}{4}\lambda_k h^2\right)u_{i_2,j_2}^h - \frac{1}{2}u_{i_2+1,j_2}^h - \frac{1}{2}u_{i_2,j_2+1}^h = \frac{1}{4}\lambda_k h^2 u_0(i_2h,j_2h),$$

$$\left(3+\frac{3}{4}\lambda_k h^2\right)u_{i_3,j_3}^h - u_{i_3-1,j_3}^h - \frac{1}{2}u_{i_3+1,j_3}^h - u_{i_3,j_3-1}^h - \frac{1}{2}u_{i_3,j_3+1}^h = \frac{3}{4}\lambda_k h^2 u_0(i_3h,j_3h).$$

At boundary points we can derive the Neumann boundary conditions from the linear equations considering the limit $h\to 0$. For example, rewriting (3.105) as

$$-\frac{u_{i_1+1,j_1}^h - u_{i_1,j_1}^h}{h} + \frac{u_{i_1,j_1}^h - u_{i_1,j_1-1}^h}{2h} - \frac{u_{i_1,j_1+1}^h - u_{i_1,j_1}^h}{2h} = \frac{1}{2}\lambda_k h\left(u_0(i_1h,j_1h) - u_{i_1,j_1}^h\right)$$

and considering $h\to 0$ yields the Neumann boundary condition $u_x = 0$.

In summary, we have a linear system which system matrix is sparse and strictly diagonally dominant as $\lambda_k > 0$. Thus, the linear system has a unique solution.

The pixel grid ($h = 1$) serves as natural grid for a spatial discretization. The numerical solution u^h is determined for all corner points of pixels. Consider a pixel in Ω_k^m with corner points (i, j), $(i-1, j)$, $(i-1, j-1)$, and $(i, j-1)$. Let $p_{i,j}$ be the center of the pixel. Then a denoised image function is defined by

$$u^h(p_{i,j})) = \frac{1}{4}\left(u^h_{i,j} + u^h_{i-1,j} + u^h_{i-1,j-1} + u^h_{i,j-1}\right). \tag{3.106}$$

Color images are denoised component-wise, i.e. each channel is treated like a scalar image.

As described in Section 3.2.8, we perform the image smoothing for $m = M$, i.e. as a postprocessing step, when the region interfaces Γ_i^M match approximately with the edges of the objects in the image. In this case, we obtain an edge-preserving image smoothing.

Curves with free endpoints

For contours with free endpoints, recall Section 3.2.10, we need to solve a discrete diffusion equation to obtain an approximation u of u_0 in each time step. The discrete diffusion equations can be derived in a similar fashion as described above. In the situation of curves with free boundaries, regions do not exist. A curve with free endpoints does not separate two different regions/segments of the image. However, similar as for interface curves, the image should also not be smoothed across the curves. Therefore, we define for $i = 1, \dots, N_x$, $j = 1, \dots, N_y$

$$A_x(i,j) = \begin{cases} h^2, & \text{if} \quad [(i-1)h, ih] \times \{j\} \cap \Gamma_{i_0}^m = \emptyset,\ \forall i_0 \in \{1, \dots, N_C\}, \\ 0, & \text{else.} \end{cases}$$

$$A_y(i,j) = \begin{cases} h^2, & \text{if} \quad \{i\} \times [(j-1)h, jh] \cap \Gamma_{i_0}^m = \emptyset,\ \forall i_0 \in \{1, \dots, N_C\}, \\ 0, & \text{else.} \end{cases}$$

and $A(i,j) = h^2$ for $i = 0, 1, \dots, N_x$, $j = 0, 1, \dots, N_y$. Thus we obtain a discrete energy (3.104), where terms are excluded in the first sum in (3.104), where an intersection with one of the curves $\Gamma_{i_0}^m$, $i_0 = 1, \dots, N_C$ occurs. Taking again the derivative with respect to $u^h_{i,j}$, we can derive linear equations for each $u^h_{i,j}$.

Automatic setting of the parameter λ

In practice, it can be difficult to choose an appropriate value for the weighting parameters λ_k, $k = 1, \dots, N_R$. Usually, the choice is dependent on the noise of the original image. For example, in case of high noise, u should not be too close to the original image u_0 and the smoothing term in the Mumford Shah energy should be weighted higher. Thus, the parameters λ_k will be chosen small if the noise is high.

In case of equal parameters $\lambda = \lambda_1 = \ldots = \lambda_{N_R}$, we propose the following method for an automatic, iterative setting of the weighting parameter: Therefore, we fix Γ and define the energies

$$\begin{aligned} E_1(u) &= \int_{\Omega \setminus \Gamma} \|\nabla u\|^2 \, \mathrm{d}x, \\ E_2(u) &= \int_{\Omega} (u - u_0)^2 \, \mathrm{d}x, \\ E^{MS,2}(u) &= E_1(u) + \lambda E_2(u). \end{aligned}$$

Let $\lambda^{(0)} > 0$ be an initial value for the weighting parameter. We set $\lambda_k = \lambda^{(0)}$ for all region indices $k \in \{1, \dots, N_R\}$ and solve the diffusion equation with Neumann boundary conditions with finite differences as described above. Let $u^{(0)}$ be the solution.

Now, let $\alpha \in (0, 1)$ be fixed chosen. The objective is to find a parameter λ and a denoised image function u such that

$$E_1(u) = \alpha E^{MS,2}(u) = \alpha(E_1(u) + \lambda E_2(u)), \tag{3.107}$$

i.e. the energy E_1 is a fixed portion (namely α) of the full energy $E^{MS,2}$.

This motivates to set the iterate $\lambda^{(1)}$ to

$$\lambda^{(1)} := \frac{(1-\alpha)E_1(u^{(0)})}{\alpha E_2(u^{(0)})}. \tag{3.108}$$

The diffusion scheme is then solved again and the solution is denoted with $u^{(1)}$. Of course, the energies E_1 and E_2 are replaced by their discrete counterparts for a practical computation, recall (3.104).

The procedure of iterative setting of $\lambda^{(i)}$ is repeated until

$$|\lambda^{(i)} - \lambda^{(i-1)}| < tol, \tag{3.109}$$

for a given tolerance value $tol > 0$ and $i \geq 1$. In this case, we end setting $\lambda = \lambda^{(i)}$ and $u = u^{(i)}$.

Concerning the computational effort, we need to solve the image restoration scheme several times for different $\lambda^{(i)}$. However, using interface curves and piecewise constant approximations during the segmentation, the restoration is performed only as post-processing step. From this point, the repeated solution of the restoration scheme until a final value for λ is found is acceptable. For contours with free endpoints, however, the restoration scheme is solved in each time step. In this case, a fixed value for λ should be chosen. Alternatively, the parameter λ can be adjusted after a certain number of time steps (like every 50 steps, for example).

3.3.8 Summary of the Image Processing Algorithm

In summing up, we propose the following algorithm for image segmentation. Given a set of polygonal curves $\Gamma^0 = (\Gamma_1^0, \dots, \Gamma_{N_C}^0)$ and $\vec{X}^0 = (\vec{X}_1^0, \dots, \vec{X}_{N_C}^0)$ with $\vec{X}_i^0(I_i) = \Gamma_i^0$, perform the following steps for $m = 0, 1, \dots, M-1$:

(i) Compute the regions Ω_k^m and the coefficients c_k^m, $k = 1, \dots, N_R$, as described in Section 3.3.6. If necessary, compute an adaptive value for σ and an adaptive time step size Δt, cf. Section 3.3.6.

(ii) Compute b^m as defined in (3.86) by using the coefficients c_k^m of step (i). Compute $\vec{X}^{m+1} = \vec{X}^m + \delta\vec{X}^{m+1}$ by solving the linear equation (3.88b), see Section 3.3.2.

(iii) Check whether topology changes occur, see Section 3.3.5. In the case of a topology change, except for a pure deletion of a curve, repeat steps (i) and (ii) n_{sub} times with a step size of τ_m / n_{sub} and execute the topology change when it occurs in a substep. After the occurrence of the topology change, the location around the involved nodes is blocked for the remaining substeps. In the next main time step, the blocking is removed.

(iv) If necessary, perform global coarsening or refinement as described in Section 3.3.6.

The blocking in step (iii) is used to prevent, for example, an alternating splitting and merging at the same location in the image. For example, after a splitting, the curves are allowed to move away from each other. A merging of the new created curves should be prevented at the location where the topology change occurred.

Having found a final segmentation of the image, a smooth approximation of the image is computed as described in Section 3.3.7.

3.3.9 Adaptations for Free Endpoints

In Section 3.2.10, we presented the analytical framework for detection of edges with free endpoints. The numerical method for image segmentation presented in this chapter can be adapted for curves with free endpoints. In case of a curve with a free endpoint, the curve is not an interface between two regions and the piecewise constant approximation cannot be used.

Step (i) and (ii) in Section 3.3.8 are replaced by computing a piecewise smooth approximation u of u_0 as described in Section 3.3.7. For F^m and thus b^m, we do not use coefficients like c_k^m since regions do not exist. Instead, we make use of the approximation u for computing F^m. Consider $i \in \{1, \dots, N_C\}$, $j \in \{j_0^i, \dots, N_i\}$. For a point which is not a free endpoint we set

$$F_{i,j}^m := \lambda \left[(u_0(\vec{X}_{i,j}^m) - u(\vec{X}_{i,j}^m + h\vec{\omega}_{i,j}^m))^2 - (u_0(\vec{X}_{i,j}^m) - u(\vec{X}_{i,j}^m - h\vec{\omega}_{i,j}^m))^2 \right], \tag{3.110}$$

with $h > 0$. As above, we use the notation $F^m = (F_1^m, \dots, F_{N_C}^m)$ with $F_i^m = (F_{i,j_0^i}^m, \dots, F_{i,N_i}^m)$.

For a free endpoint, we compute $\delta X_{i,j}^{m+1}$ by using discrete versions of (3.70), (3.72), (3.74) and (3.75). We introduce the tangential vectors $\vec{\tau}_{i,0}^m = (\vec{X}_{i,1}^m - \vec{X}_{i,0}^m)/h_{i,\frac{1}{2}}$, and $\vec{\tau}_{i,N_i}^m = (\vec{X}_{i,N_i}^m - \vec{X}_{i,N_i-1}^m)/h_{i,N_i-\frac{1}{2}}$. We approximate (3.70) by

$$\frac{1}{\tau_m} \delta \vec{X}_{i,0}^{m+1} \cdot \vec{\tau}_{i,0}^m = \sigma - \mu \left[|\vec{\tau}_{i,0}^m \cdot \vec{e}_1| (\nabla_h^2 u(\vec{z}^{0,1}))^2 + |\vec{\tau}_{i,0}^m \cdot \vec{e}_2| (\nabla_h^1 u(\vec{z}^{0,2}))^2 \right], \tag{3.111}$$

Similarly, (3.72), (3.74) and (3.75) are approximated by discrete versions, replacing $\vec{\nu}$ and $\vec{\tau}$ by $\vec{\omega}_{ij}^m$ and $\vec{\tau}_{i,j}^m$, respectively, where $j \in \{0, N_i\}$ is the index of the free endpoint.

The main effort of this method compared to the Chan-Vese method for interface curves is that we have to solve a two-dimensional diffusion equation several times during the segmentation; not only as a postprocessing step. In the experiments described in Section 3.4, we solve the bulk equation only every 10th iteration steps and use the image approximation for the next 10 curve evolution steps.

In principle, topology changes can be detected similarly as for interface curves by using an artificial background grid. In addition to the topology changes discussed in Section 3.3.5, topology changes involving the free endpoints can occur. If two free endpoints of one curve are located in one square of the background grid, an open contour becomes a closed contour. If two free endpoints of two different curves meet, the two curves merge to one single curve, and the former free endpoints become inner nodes of the new curve. If a free endpoint and an inner point of a curve meet, a triple junction is created.

As an alternative, we can start the segmentation using interface curves and the Chan-Vese method with piecewise constant approximations. As a postprocessing step, we can consider the derivatives of the image function in normal direction at the final curves (or the jump of the image function across the curves). We replace interface curves by curves with free endpoints if the derivatives in normal direction are locally very small. For that, we delete those parts of a curve where the derivative is small which results in curves with free endpoints. Next, we compute some steps of the segmentation method with free endpoints to obtain the final contours.

Topology changes occur only in rare cases when using a postprocessing evolution of curves with free endpoints. In most situations, topology changes were already detected in the previous evolution. By applying the proposed technique it is possible to easily handle interface curves and non-interface curves in one image.

3.4 Results

We apply the image segmentation and restoration method of Sections 3.2 and 3.3 on artificial and real, gray-scaled and color images.

3.4.1 Implementation

The algorithms are implemented in MATLAB and tested with version MATLAB R2011b.

Some functions are written in C as so-called *mexFunctions* and can be called from MATLAB routines. For example, a function for computing the regions (initial regions where all pixels are affected, and update of the regions locally around the curves) and a function for evaluating the external forcing term are written as mexFunctions.

The UMFPACK algorithm as MATLAB built-in routine is used to solve the linear equation (3.88b), see also Davis (2004).

The software can be configured using a parameter file. Internal, structures are used for keeping information of the curve (structure *Gamma*), of the regions (structure *Omega*) and of the image (structure *Image*).

3.4.2 Artificial Test Images

Two-phase image segmentation and restoration with noise and topology changes

In the first example, we consider a gray-scaled, noisy image showing the letters A, B and C. Figure 3.13 shows the result of a segmentation of this image in two regions. The contours at several time steps and the corresponding piecewise constant approximation given by (3.19), i.e. $u_{|\Omega_k} = c_k$, $k = 1, 2$, are presented. The gray value in the region Ω_k, $k = 1, \ldots, N_R$ (here $N_R = 2$), is the mean of u_0 in Ω_k, recall (3.23).

In the first segmentation of this image, we use one single, closed initial curve placed in the middle of the image which evolves in time according to the evolution equation (3.35a). For the segmentation, we use the weighting parameters $\sigma = 1$ and $\lambda = 20$ and the time step size $\Delta t = \tau_m = 0.1$. This example demonstrates the general behavior of the algorithm. In particular, it demonstrates the detection of some topology changes. Here, the curve is split up

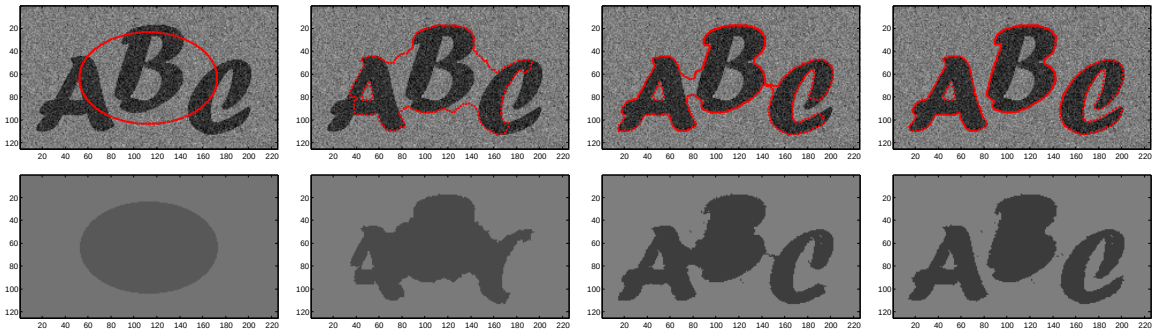


Figure 3.13: Image segmentation of a noisy gray-scaled image with multiple objects to be detected. Test 1: interior edges of the letter B are not detected. Parameters: $\Delta t = 0.1$, $\lambda = 20$, $\sigma = 1$. First row: Original image and contours for $m = 1, 240, 450, 600$. Second row: Piecewise constant approximation.

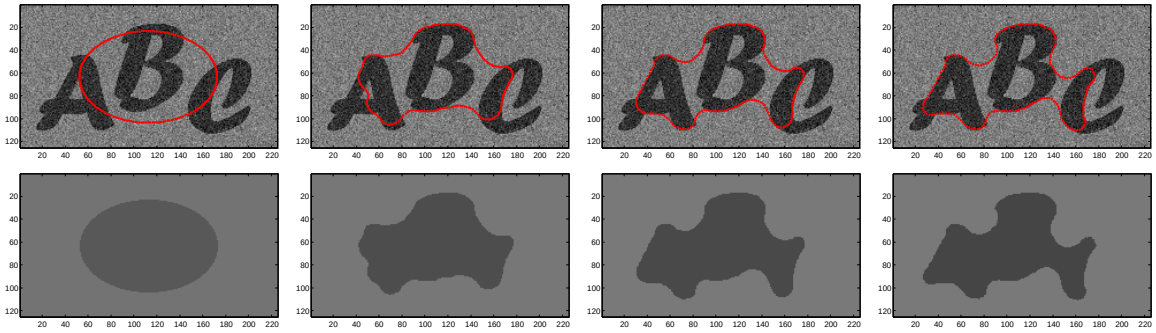


Figure 3.14: Image segmentation of a noisy gray-scaled image with multiple objects to be detected. Test 2: choice of a too large weight for the curvature term. Parameters: $\Delta t = 0.1$, $\lambda = 20$, $\sigma = 10$. First row: Original image and contours for $m = 1, 300, 600, 1000$. Second row: Piecewise constant approximation.

into four subcurves (two enclosing the letter A, one each for letter B and C). This experiment also presents limitations of the segmentation when using only one single initial curve: The interior contours of the letter B are not detected as they are enclosed by the initial contour.

In the second experiment, we increase the parameter σ which weights the curvature term in (3.35a) to $\sigma = 10$ while keeping $\lambda = 20$ as in the first experiment. Figure 3.14 shows that the curve is not split up since the large weight of the curvature term tries to keep the length of the curve as small as possible. Finally, in a third experiment, we use two initial contours and reset σ to 1, see Figure 3.15. In this case, even the initial contours of B are detected. Merging and splitting of curves appear as topology changes.

Figure 3.16 compares the final piecewise smooth approximations of the image, cf. Section 3.2.8 and 3.3.7, executed after the last iteration step. The partial differential equation $-\frac{1}{\lambda}\Delta u + u = u_0$ with Neumann boundary conditions is solved on each phase separately. The initial contours of B, which are not detected in the first experiment, are smoothed out, whereas the detected boundaries remain sharp, see Figure 3.16 (left). Since the segmentation has failed in the second experiment, the result of the image denoising is consequently quite bad. In the third experiment, all edges and regions have been detected with success. Consequently, all edges remain sharp in Figure 3.16 (right).

It has been demonstrated that the segmentation result is strongly dependent on the choice

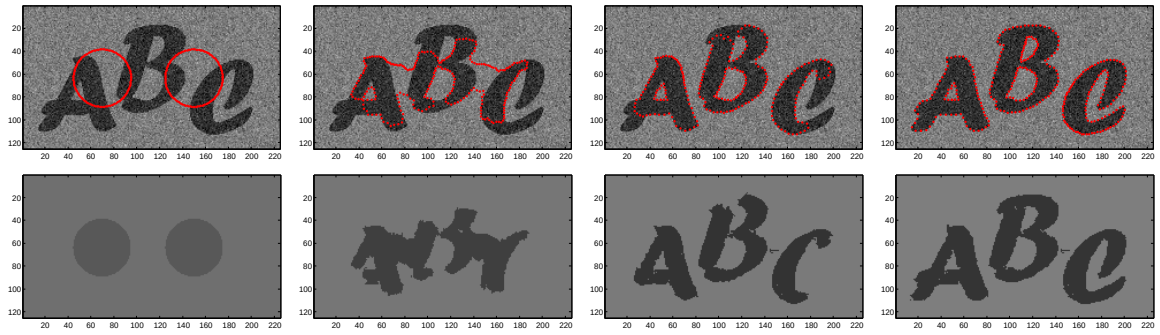


Figure 3.15: Image segmentation of a noisy gray-scaled image with multiple objects to be detected. Test 3: use of two initial curves, detection also of the interior edges. Parameters: $\Delta t = 0.1$, $\lambda = 20$, $\sigma = 1$. First row: Original image and contours for $m = 1, 180, 300, 600$. Second row: Piecewise constant approximation.

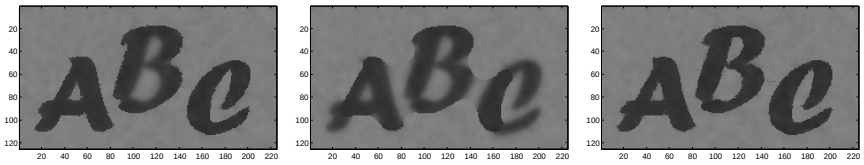


Figure 3.16: Comparison of the final smooth approximation with $\lambda_k = 1$, $k = 1, 2$, Test 1-3 (left to right).

of the weighting parameters and the number and location of the initial curves. From this experience, we can derive two recommendations:

- As it may be difficult to choose an appropriate value for σ in advance, one can make use of the adaptive setting of the parameter as described in Section 3.3.6 and can control the perimeter energy with respect to the total energy.
- The higher the number of initial contours, the more regions and consequently the more details can be detected. For real images, see Section 3.4.3, we always use several initial curves.

Multiphase demonstration and adaptive grid size for topology changes

As a second test image, a gray-scaled image with three objects is segmented in multiple phases, see Figure 3.17. The first row in Figure 3.17 shows the given image and the contours for four different iteration steps. The second row shows the piecewise constant, approximating image given by (3.19). Having detected the objects, the contours match with the edges of the objects, and the approximating image is a smooth, piecewise constant version of u_0.

This sample image demonstrates two kinds of topology changes: splitting of a curve and creation of boundary intersection points. The green curve splits into two single subcurves. At time step $m = 560$, two parts of the curve nearly touch. The splitting is detected as described in Section 3.3.5. The red contour in Figure 3.17 intersects the image boundary $\partial\Omega$ at two positions. Two open curves, each with two boundary intersection points, exist at time step $m = 2150$. The length of the smaller subcurve decreases continuously in the following time steps, and the curve is deleted close to the upper left corner of the image. The remaining curve is attracted to the boundary of the upper left gray square.

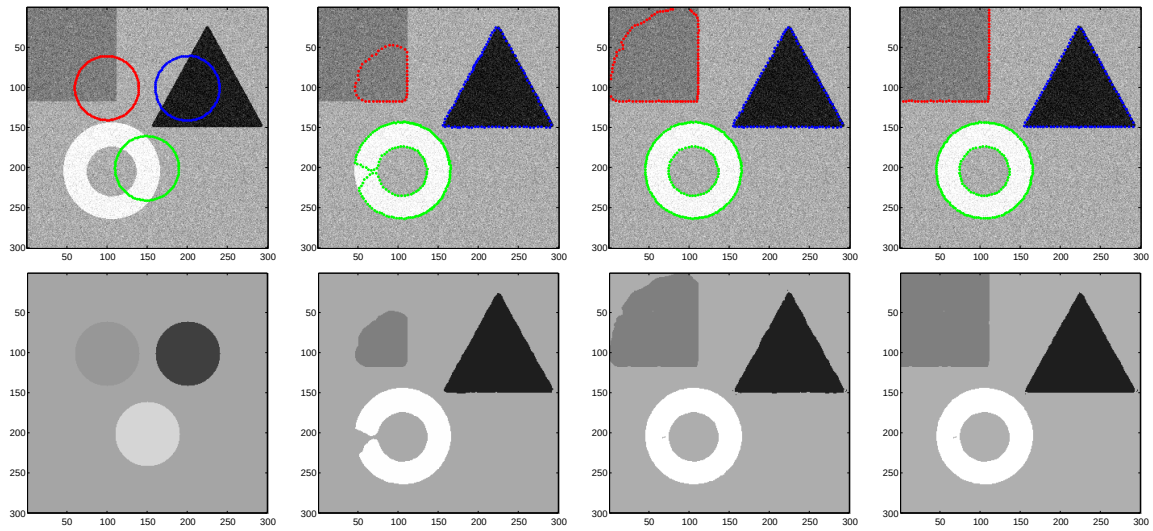


Figure 3.17: Multiphase image segmentation of a gray-scaled image with multiple objects to be detected, with $\Delta t = 0.1$, $\lambda = 20$, σ-factor 20%. First row: Original image and contours for $m = 1, 560, 2150, 3000$. Second row: Piecewise constant approximation.

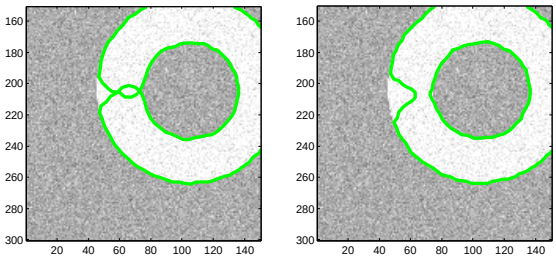


Figure 3.18: Effect of the size a of the background grid used to detect topology changes. Left: Detection of the topology change fails when using a fixed grid size $a = 1$. Right: Successful detection when using an adaptive grid size $a = 6.12$. In the algorithm, an adaptive choice of a makes sure that a situation like the one in the left image does not occur in practice.

In this example, we set the size of the auxiliary grid used to detect topology changes adaptively with respect to the position change of the nodes. The adaptive setting of the grid size makes the algorithm robust with respect to larger position deviations. Therefore, we segment the image again using a time step size of $\Delta t = 0.25$ and compare the results using a constant grid size and an adaptive grid size. Figure 3.18 presents an excerpt of the image and the affected curve shortly after a topology change. In the left subfigure, the change is not detected since a too small fixed grid size of $a = 1$ is used. In the right subfigure, we compute a by setting $a_0 := 2\max\{\|\delta X_{i,j}^{m+1}\|, i = 1, \ldots, N_C, j = j_0^i, \ldots, N_i\}$ and $a = \max\{0.25, \min\{a_0, 10\}\}$, i.e. we use a lower limit of 0.25 pixels and an upper limit of 10 pixels for the grid size. The splitting of the green curve is detected in the second experiment, cf. Figure 3.18 (right). At the time when the splitting is detected, $a = 6.12$ is used. Motivated by this experiment, we apply the adaptive setting of a for each segmentation presented in this section.

The Mumford-Shah energy (3.20) for the piecewise constant approximations u is shown in Figure 3.19. In addition to the total energy, the perimeter energy $\sigma|\Gamma|$ is plotted to visualize the proportion of the length term which ensures smoothness of the curve. The energy decreases quickly at the beginning. After approximately 600 steps, the ring and the

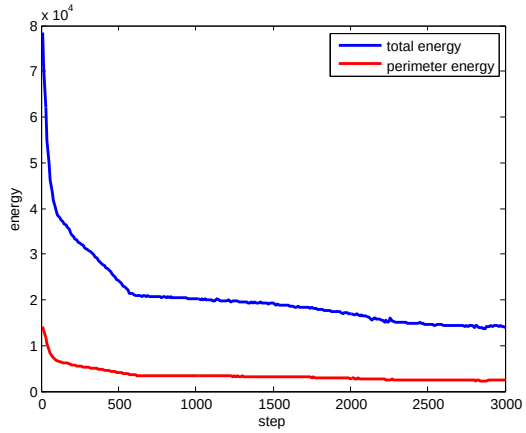


Figure 3.19: Energy decrease during the segmentation of the test image of Figure 3.17.

triangle are both detected. The difference between the gray value of the square and the gray value of the background is small. Therefore, the segmentation of the square is much slower compared to the segmentation of the black triangle and the white ring. In the next example, we present how the adaptive time step control can decrease the number of necessary evolutions steps.

Color images and time step control

Figure 3.20 presents the segmentation results of applying the algorithm on a color image. For this experiment, the RGB space is used, and for all color channels $j \in \{1, 2, 3\}$ the parameter $\lambda_j = 5$ is used. When two different curves touch, an intermediate quadruple junction occurs. A new small curve is created and the quadruple junction is replaced by two triple junctions connected by the new contour, as illustrated in Figure 3.10.

The Mumford-Shah energy and the perimeter energy during the evolution are shown in Figure 3.21. The segmentation is finished after 300 time steps. In this test image the colors of the different objects differ significantly. Therefore, the regions are easy to detect.

We repeat the segmentation of the image with the colored balls to demonstrate the adaptive control of the time step size (see Section 3.3.6). Every 10 iterations, we compute the external energy and update the time step size. We increase the time step size if the relative external energy compared to the step $m - 10$ is smaller than $\Delta E_{\min} = 0.05$ and decrease the time step size, if the energy decrease is larger than $\Delta E_{\max} = 0.1$. Figure 3.22 shows the relative energy and the adaptive time step size as well as the final segmentation. As initial time step size, we use a small time step of $\Delta t = 0.025$. Since the energy decrease is too small, the time step size is increased several times by a factor of 2. Around $m = 40 - 50$, the change in the energy is too large and the time step size is decreased. At $m = 120 - 130$, one can observe a smaller energy decrease. Consequently the time step is increased again to speed up the computation. At $m = 180$, the relative energy is smaller than $\Delta E_{\text{del}} = 0.005$ and the segmentation is stopped. Also with adaptive time step sizes, the detection of topology changes has been performed with success, since the grid size a used by the detection algorithm is set adaptively dependent on the position change of the mesh points.

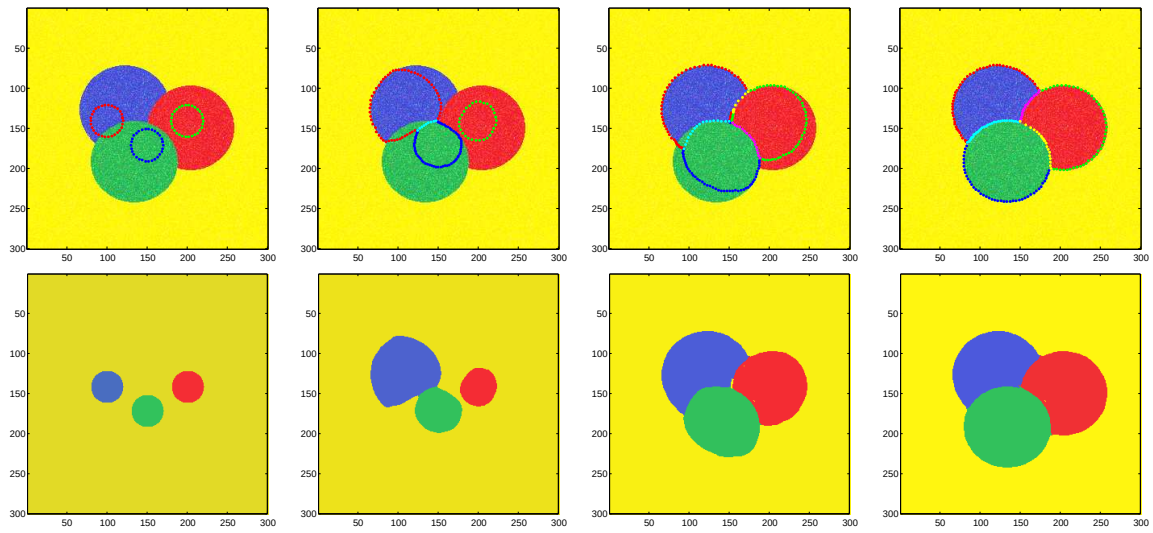


Figure 3.20: Multiphase image segmentation of a color image using RGB space, with $\Delta t = 0.1$, $\lambda_1 = \lambda_2 = \lambda_3 = 5$, σ-factor 25%. First row: Original image and contours for $m = 1, 100, 200, 300$. Second row: Piecewise constant approximation.

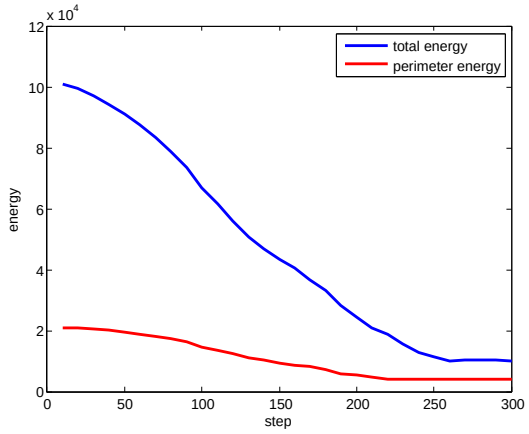


Figure 3.21: Energy decrease during the segmentation of the test image of Figure 3.20.

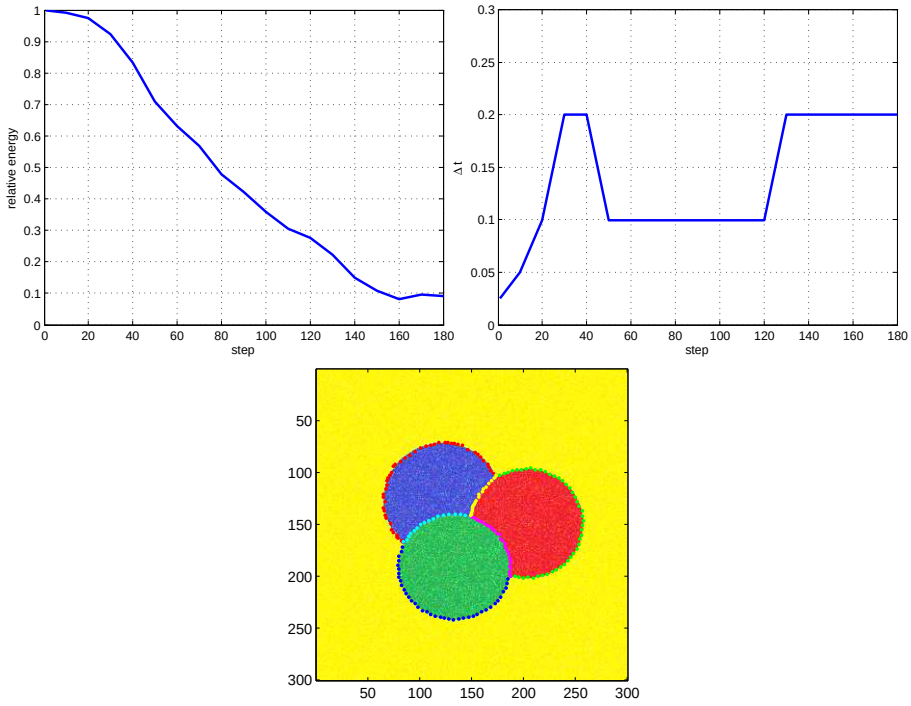


Figure 3.22: Adaptive setting of the time step size Δt dependent on the energy decrease. First row: relative external energy (left) and time step size (right). Second row: Final segmentation.

Image restoration

In Section 3.2.8, we proposed a method for image smoothing. The partial differential equation $-\frac{1}{\lambda}\Delta u + u = u_0$ with Neumann boundary conditions is solved on each phase separately. Having previously detected the regions, the image smoothing is performed as a postprocessing step. Figure 3.23 (far left) shows a noisy image with five different colored stripes. The brightness increases from left to right. The second subfigure from the left shows the result solving the diffusion equation on Ω neglecting the previously detected regions. The edges are strongly smoothed out.

This motivates a search for an approximation of the image which enhances the edges. It is necessary to use a piecewise smooth approximation since a piecewise constant approximation would be a too strong simplification, and the brightness change of the original image would get lost. The smoothing effect severely depends on the parameter λ: If λ is large, $\frac{1}{\lambda}$ is small and the approximation u is close to the original image u_0. The smaller λ is, on the contrary, the bigger is the smoothing effect. Therefore, choosing $\lambda = 1$, the noise is not completely smoothed out, cf. the third subfigure in Figure 3.23. Choosing $\lambda = 0.1$ results in a smooth approximation u of u_0, cf. fourth subfigure. The change in the brightness from left to right is still conserved. By solving the diffusion equation with Neumann boundary conditions separately in each phase, the edges remain sharp.

In Section 3.3.7 we discussed a possible iterative method for automatic setting of the parameter λ. In the example shown by Figure 3.24 and 3.25, the energy $E_1 = \int_{\Omega\setminus\Gamma} \|\nabla u\|^2 \,\mathrm{d}x$ should be 10% of the full energy $E_1 + \lambda E_2$, i.e. $\alpha = 0.1$ (using the notation of Section 3.3.7). If $E_2 = \int_\Omega \|u - u_0\|^2 \,\mathrm{d}x$ is small, the approximation u will be close to the original image u_0. The energy E_1 provides that u is sufficiently smooth (here: 10% of the full energy).

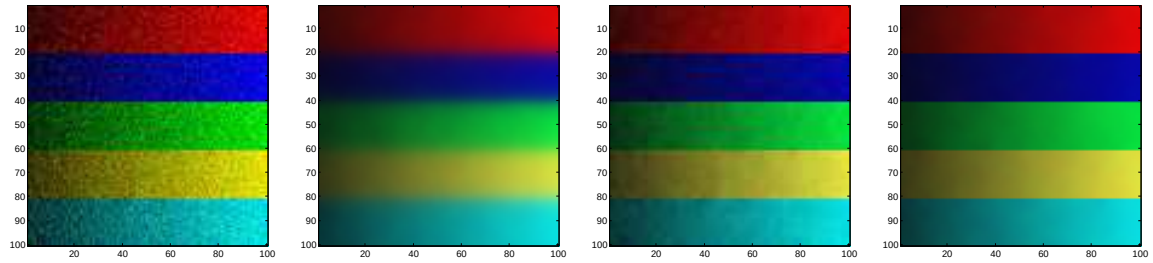


Figure 3.23: Test of the image denoising method. Images left to right: Original image, image denoising without edge enhancement with $\lambda = 0.1$, image denoising result with edge enhancement with $\lambda = 1$ and $\lambda = 0.1$.

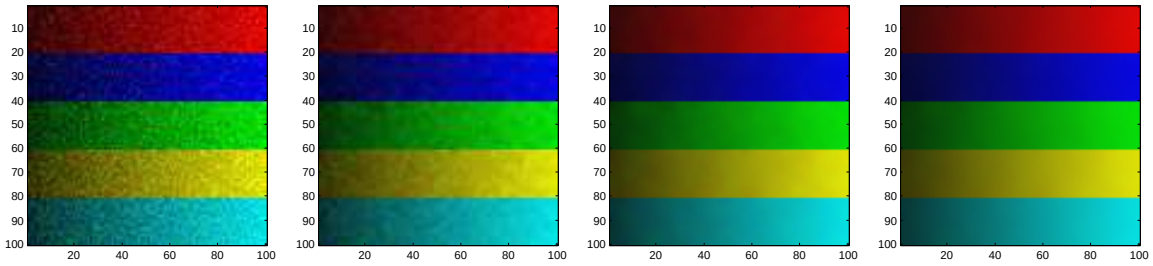


Figure 3.24: Test of the image denoising method with an iterative setting of λ, portion of E_1: 10%. Images left to right: Original image, approximation iteration 1 ($\lambda = 10$), iteration 4 ($\lambda = 0.1647$), iteration 8 ($\lambda = 0.0562$).

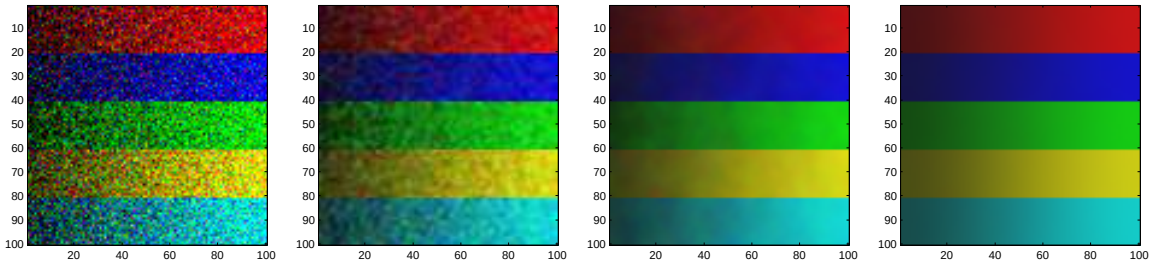


Figure 3.25: Test of the image denoising method with an iterative setting of λ, portion of E_1: 10%. Images left to right: Original image, approximation iteration 1 ($\lambda = 10$), iteration 4 ($\lambda = 0.0896$), iteration 7 ($\lambda = 0.0060$).

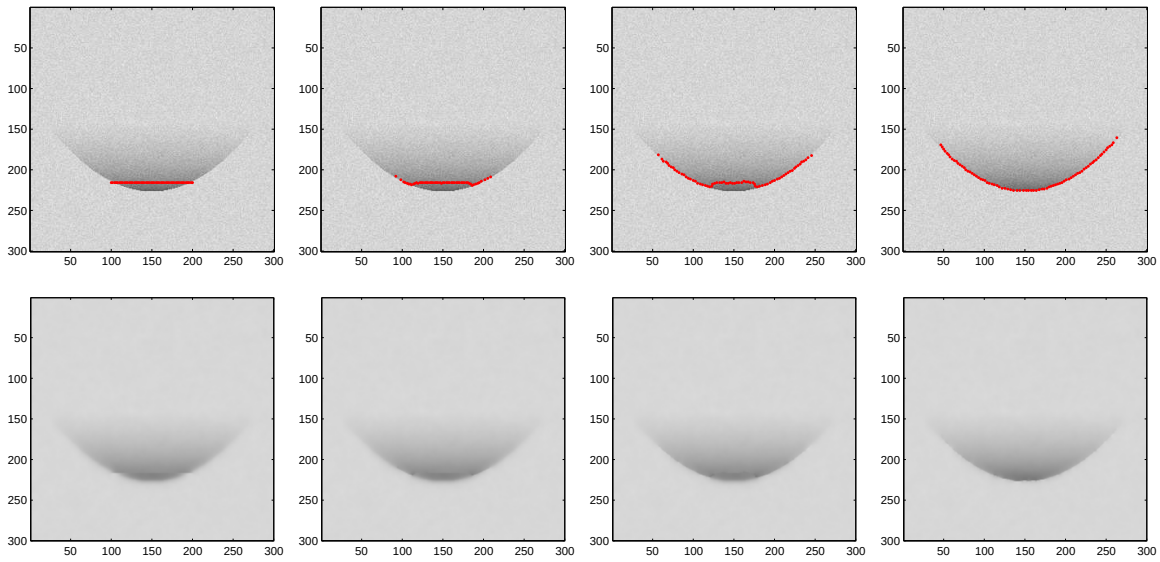


Figure 3.26: Contour with free endpoints. Contour detection and image denoising result. First row: Original image and contours for $m = 1, 100, 1000, 6000$ using $\Delta t = 0.032$, $\sigma = 1$, $\lambda = 100$ and $\mu = 50000$. Second row: Denoised image.

Figure 3.24 and Figure 3.25 each show a noisy image (original image), the result of the first iteration (starting with $\lambda = 10$), an intermediate iteration and the result of the final iteration step. The iteration is stopped, when the difference between the current parameter λ and the previous value is less than 0.001. In the experiment shown by Figure 3.24, the final parameter is $\lambda = 0.0562$. Since the noise in the original image presented by Figure 3.25 is higher, λ has to be set to a smaller value. For the second example, the final value for λ is 0.0060, i.e. one magnitude smaller compared to the example presented in Figure 3.24.

Curves with free endpoints

The handling of free endpoints as described in Section 3.2.10 and Section 3.3.9 is demonstrated by some experiments. Figure 3.26 shows an example image, where the contour Γ is a non-interface curve, i.e. it does not separate two different regions of the image. The figure presents the results of image segmentation and denoising. It can be observed that the image is not smoothed out across Γ. Further a growth of the curve in tangential direction can be observed. The growth stops when the inequalities (3.71) and (3.73) become equalities. This depends on the absolute values of the difference quotients $|\nabla_h^i u|$, $i = 1, 2$, and the weighting parameters σ and μ. For $\sigma = 1$, we have to choose a high value for μ: The image approximation u attains values in $[0, 1]$. Typically, differences of the form $u(\vec{x} + h\vec{e}_i) - u(\vec{x})$ are of magnitude 10^{-2}. Since $\Omega = [1, 300] \times [1, 300]$ and $h = 1$, $|\nabla_h^i u|^2$ is of magnitude 10^{-4}. Therefore, we use $\mu = 5 \cdot 10^4$. If we used a normalized image domain $\Omega = [0, 1] \times [0, 1]$, the pixel grid would have a grid size of $h = 1/300$ and $h^2 = 1/90000$. In this case, we would choose a weight between 0.1 and 1 for the parameter μ.

In a second experiment, we study a crack tip problem which has also been considered by Pock et al. (2009b). The image function is given by

$$u_0(\vec{x}) = a\sqrt{r(\vec{x})}\sin(\theta(\vec{x})/2) + b, \tag{3.112}$$

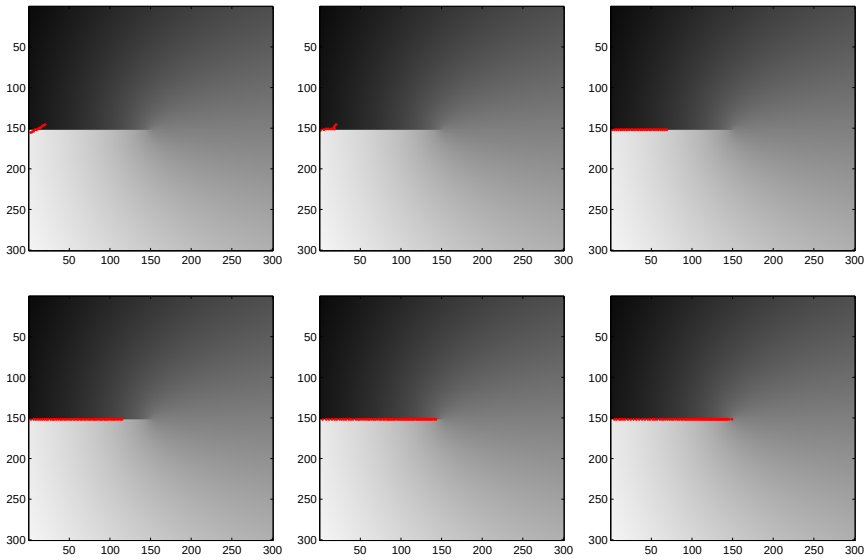


Figure 3.27: Image segmentation and edge detection result of an image with a free endpoint, see also Pock et al. (2009b). Original image and contours for $m = 1, 50, 500, 1000, 2000, 3000$ (row-wise) using $\Delta t = 0.001$, $\sigma = 1$, $\lambda = 100$ and $\mu = 50000$.

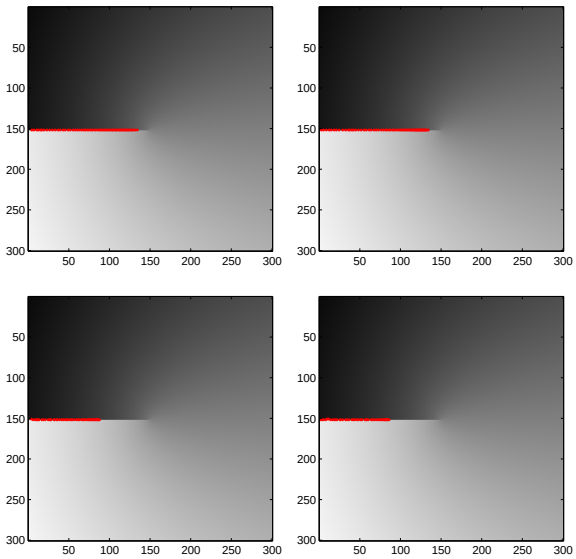


Figure 3.28: Image segmentation and edge detection result of an image with a free endpoint, see also Pock et al. (2009b). Original image and contours for $m = 3000$ (left) and $m = 5000$ (right) using $\sigma = 100$ (first row) and $\sigma = 500$ (second row), $\Delta t = 0.001$, $\lambda = 100$ and $\mu = 50000$.

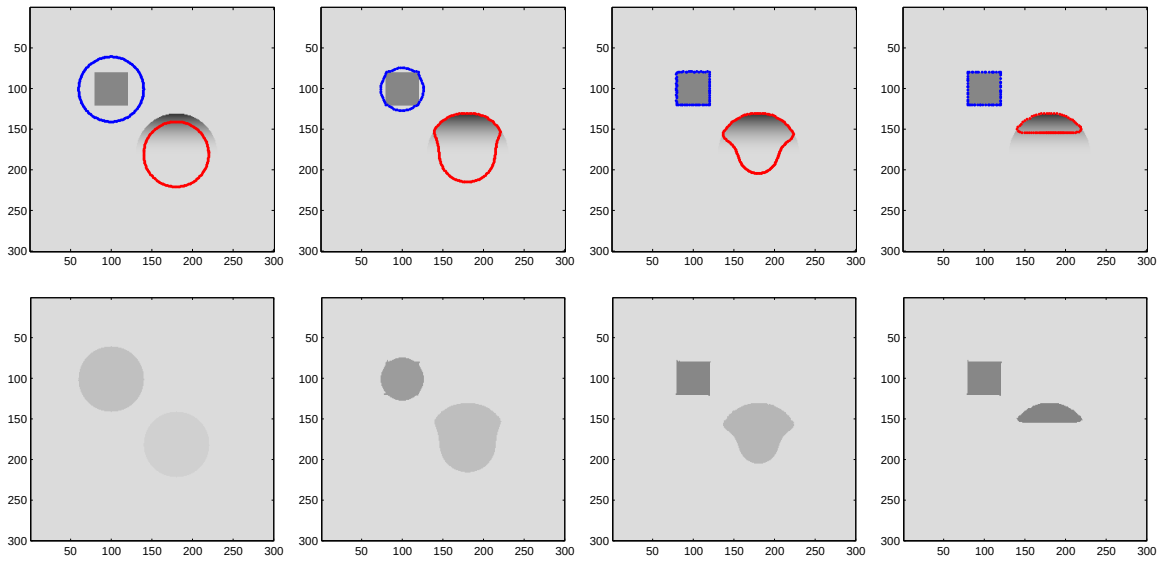


Figure 3.29: Image segmentation using Chan-Vese and piecewise constant approximation. First row: Original image and contours for $m = 1, 500, 1000, 3000$ using $\Delta t = 0.1$, $\sigma = 1$, $\lambda = 10$. Second row: Piecewise constant approximation.

where $r(\vec{x}) \geq 0$, $\theta(\vec{x}) \in (-\pi, \pi]$ are polar coordinates with $r = 0$ corresponding to the image center, and $a, b \in \mathbb{R}$ are constants such that u_0 has values in $[0, 1]$.

Figure 3.27 shows the evolution of a contour with one free endpoint. The second endpoint belongs to the image boundary. At time step $m = 3000$, the free endpoint is located at the image center and the curve matches with the edge in the image. As discussed in Section 3.2.10 (see Equation (3.73)), the parameter σ, which weights the length term, needs to be chosen small enough such that the curve can extend. If σ and μ are fixed, the absolute value of the difference quotients must be large enough such that the length of the curve increases. In this example, the edge is a horizontal line and the position where the curve stops depends on the value of $\nabla_h^2 u$, i.e. on the difference quotient in y-direction.

We rerun the example using $\sigma = 100$ and $\sigma = 500$ instead of $\sigma = 1$. Figure 3.28 shows the results at time step $m = 3000$ and $m = 5000$. In both cases, the free endpoint does not reach the center of the image since the value of σ has been set larger. The growth of the curve already stops at larger values of $\nabla_h^2 u$, recall conditions (3.62) and (3.73)). The results at time step $m = 5000$ show that there is no significant motion between $m = 3000$ and $m = 5000$.

In another experiment, which demonstrates the evolution of curves with free endpoints, we first apply the Chan-Vese algorithm using two interface-curves, i.e. we first segment a given image in three regions, see Figure 3.29. In a postprocessing step, we delete those nodes where the jump of u_0 across the curve is smaller than a given tolerance. First, we used $tol = 0.1$ as tolerance to delete nodes. As a result, the red curve becomes an open curve with two free endpoints. Figure 3.30 shows the results of a postprocessing evolution of the curve. This example shows that our methods for image segmentation and denoising can be applied also on images with both open and closed edges.

Table 3.1 shows the values of the discrete Mumford-Shah energy (3.60) for the last step of the Chan-Vese piecewise constant segmentation with closed region boundaries (cf. Figure 3.29, $m = 3000$) and for the initial and final step of the postprocessing with one open boundary (cp. Figure 3.30, $m = 1$ and $m = 400$). Note, that the absolute values are large,

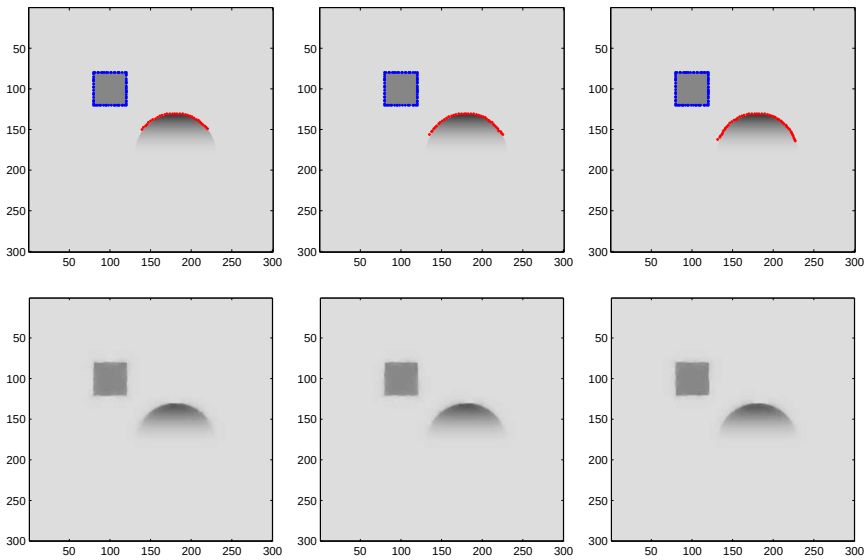


Figure 3.30: Postprocessing evolution with a contour with two free endpoints using $tol = 0.1$ to obtain the initial contour. Image segmentation and denoising result. First row: Original image and contours for $m = 1, 100, 400$ using $\Delta t = 0.05$, $\sigma = 1$, $\lambda = 10$ and $\mu = 30000$. Second row: Denoised image.

Processing Method	Step Nr	Discrete Mumford-Shah Energy
Chan-Vese, piecewise constant	3000 (final)	22364.94
Postprocessing, free endpoints	1 (start)	23162.04
Postprocessing, free endpoints	400 (final)	18284.36

Table 3.1: Comparison of discrete Mumford Shah Energy of the last step of the Chan-Vese method, the first step of the postprocessing with a curve with free endpoints and the last step of the postprocessing.

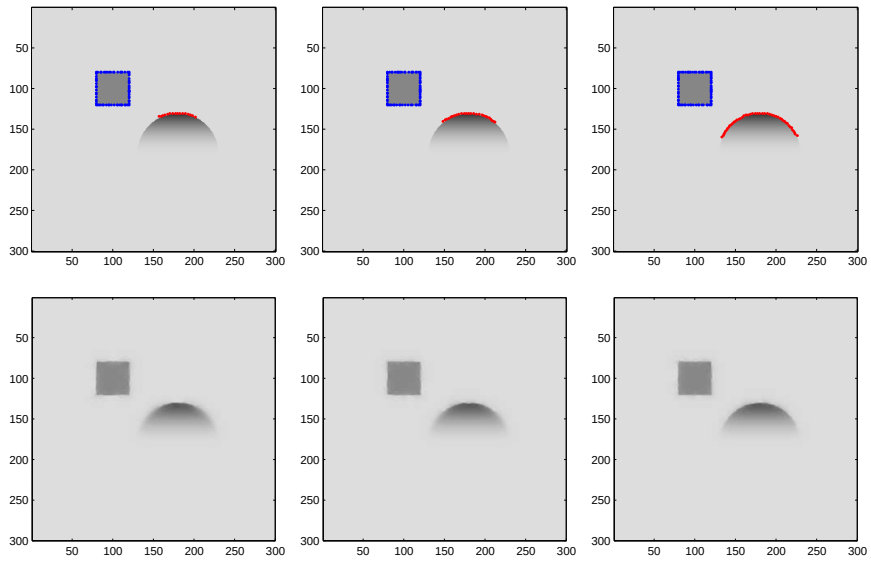


Figure 3.31: Postprocessing evolution with a contour with two free endpoints using $tol = 0.5$ to obtain the initial contour. Image segmentation and denoising result. First row: Original image and contours for $m = 1, 100, 800$ using $\Delta t = 0.05$, $\sigma = 1$, $\lambda = 10$ and $\mu = 30000$. Second row: Denoised image.

since the image consists of 90000 pixels and the size of each pixel is 1×1. The average contribution of each pixel to the energy is < 1; however, it sums up to a value of magnitude $2 \cdot 10^4$.

From Table 3.1, we can observe that the energy even slightly increases from the last step of the Chan-Vese piecewise constant method to the first step of the postprocessing evolution. Deleting part of the curve does *not* decrease the energy in this example. However, at the end of the postprocessing evolution with one open contour, we obtain a decrease of the energy. The energy at $m = 400$ of the postprocessing is 81.75% of the energy of the last step of the piecewise constant segmentation with closed boundaries. Therefore, if we delete part of the curve and if we let the curve with free endpoints evolve again, we will obtain a final curve such that the corresponding discrete Mumford-Shah energy (3.60) is reduced compared to the Chan-Vese piecewise constant result.

Next, we investigate the influence of the tolerance value *tol*, which is used for the deletion of some nodes of the curve. Note, that the image function has values in $[0, 1]$ where 0 corresponds to black and 1 corresponds to white color. The exact value of the tolerance *tol* influences only the start curve of the second curve evolution. We repeat the postprocessing evolution and use $tol = 0.5$ as tolerance resulting in a different initial curve.

Figure 3.31 shows the postprocessing evolution of the curve. Of course, since the initial contour in Figure 3.31 (1st column) is smaller compared to the initial contour in Figure 3.30, more iterations steps are needed to obtain the final contour.

The final result is independent on the exact initial curve as long as it is of the same type, i.e. open with free endpoints; not closed or not fully deleted. In our example the largest jump of u_0 across the final red curve of the first evolution (cf. 4th column in Figure 3.29) is 0.61, the smallest jump is 0.05. The large difference between the largest and smallest jump can be used as an indicator to replace the interface-curve by a curve with two free endpoints. (On the contrary, the jump across the blue curve is constant in this example.) As tolerance

value *tol* any value larger than 0.05 and smaller than 0.61 could be chosen. For $tol \leq 0.05$ no node point would be deleted resulting in an unchanged curve. For $tol \geq 0.61$ all nodes and therefore the entire curve would be deleted. All values between the two thresholds can be theoretically used. Therefore, in this example, the final result is independent on the exact value of *tol* as long it is in $(0.05, 0.61)$.

3.4.3 Real Images

In addition to artificial test images, the segmentation and restoration technique is applied on real images. We present how the developed methods for multiphase segmentation, detection and execution of topology changes, contours with free endpoints and image restoration by piecewise smooth functions work with real images.

Segmentation of a gray-scaled, medical image

Figure 3.32 presents the segmentation of a medical image. It shows the original image and the segmentation for $m = 1, 100, 500$ and 1500 (first row) and the piecewise constant approximation (second row). This example demonstrates the creation of new interfaces and triple junctions. The parameter λ, which weights the external forcing term, is chosen high ($\lambda = 400$), as the brightness differences of some objects and the background are small. The σ-factor is set to 5%. Additionally, lower limits of $\sigma_{\min} = 15$ for $m < 400$ and $\sigma_{\min} = 5$ for $m \geq 400$ are applied. A magnification of the final segmentation demonstrates the ability of region-based methods to handle weak edges (third row, left). Figure 3.32 additionally presents the final piecewise smooth approximation for different weighting parameters $\lambda = 1$ and $\lambda = 23.115$ (third row, subfigures 2-3). The latter value is determined by the routine for the automatic setting of the parameter λ, see Section 3.3.7.

A contour which partly represents a weak edge can also be replaced by a curve with free endpoints. Figure 3.33 and 3.34 show an excerpt of the medical image and the result of edge detection. We first use the Chan-Vese algorithm with piecewise constant image approximation for segmenting the image, see Figure 3.33. In this first segmentation step, also topology changes (boundary contacts) occur. After the prior segmentation, part of the red curve is deleted (using a tolerance of 0.1 for the jump across the curve) resulting in a curve with free endpoints. Figure 3.34 shows the result of the postprocessing evolution. Small tangential motions of the free endpoints can be observed.

Segmentation of color images

Figure 3.35 shows the result of segmenting a color image from the Caltech 101 dataset (`http://www.vision.caltech.edu/feifeili/Datasets.htm`) (Fei-Fei et al., 2004). For this experiment, the chromaticity-brightness color space is used. The weighting parameters for the external forcing terms are set to $\lambda_C = 80$ and $\lambda_B = 20$ such that the chromaticity has a higher influence on the region-based segmentation. The ability to handle multiple phases, topology changes (i.e. boundary intersection and creation of triple junctions) and vector-valued image data is demonstrated in this example.

A second real colored image showing two flowers, from the Berkeley Segmentation Data Set and Benchmarks 500 (BSDS500) (`http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/resources.html`) (Arbelaez et al., 2011; Martin et al., 2001)), is used

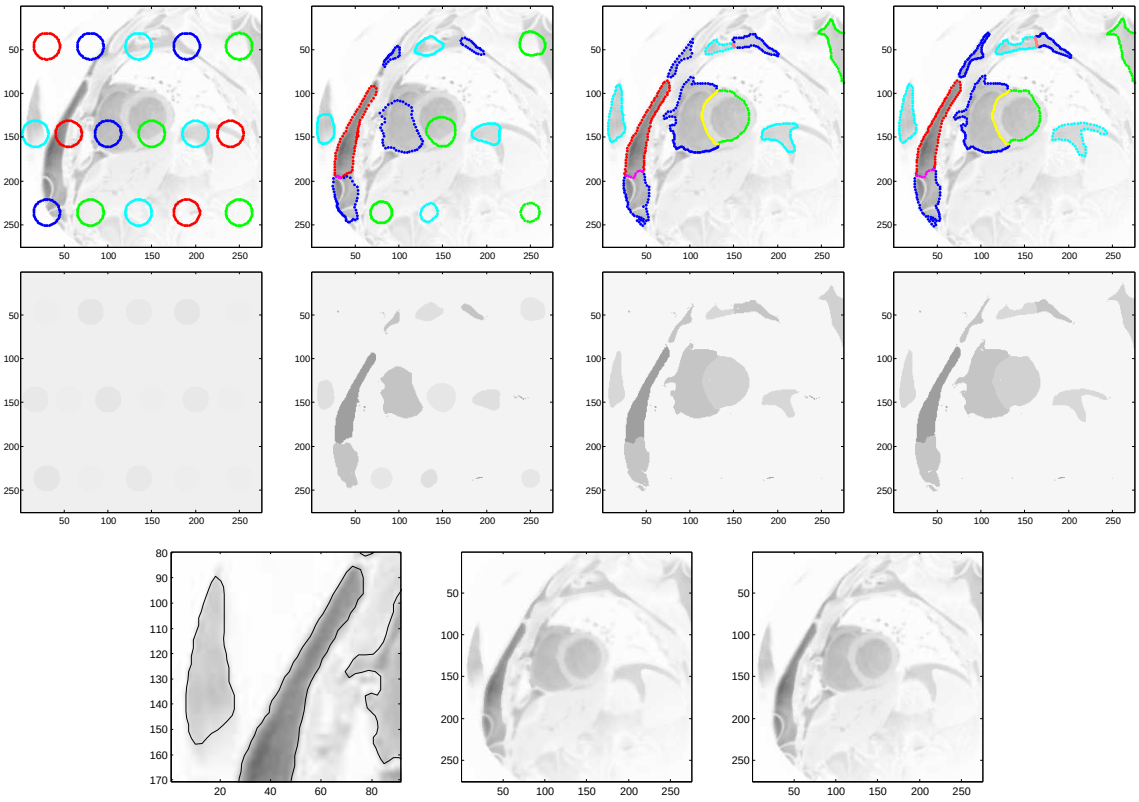


Figure 3.32: Multiphase image segmentation of a medical image, with $\Delta t = 0.02$, $\lambda = 400$, σ-factor 5%. First row: Original image and contours for $m = 1, 100, 500, 1500$. Second row: Piecewise constant approximation. Third row: Magnification of the final segmentation to demonstrate a weak edge, approximation u computed in the postprocessing step for $\lambda = 1$ (initial value) and $\lambda = 23.115$ (final value after 11 iterations, automatic computing of λ, portion of E_1: $\alpha = 0.2$). Image courtesy: Dr. Declan O'Regan and the Robert Steiner MR Unit, MRC Clinical Sciences Centre, Imperial College London.

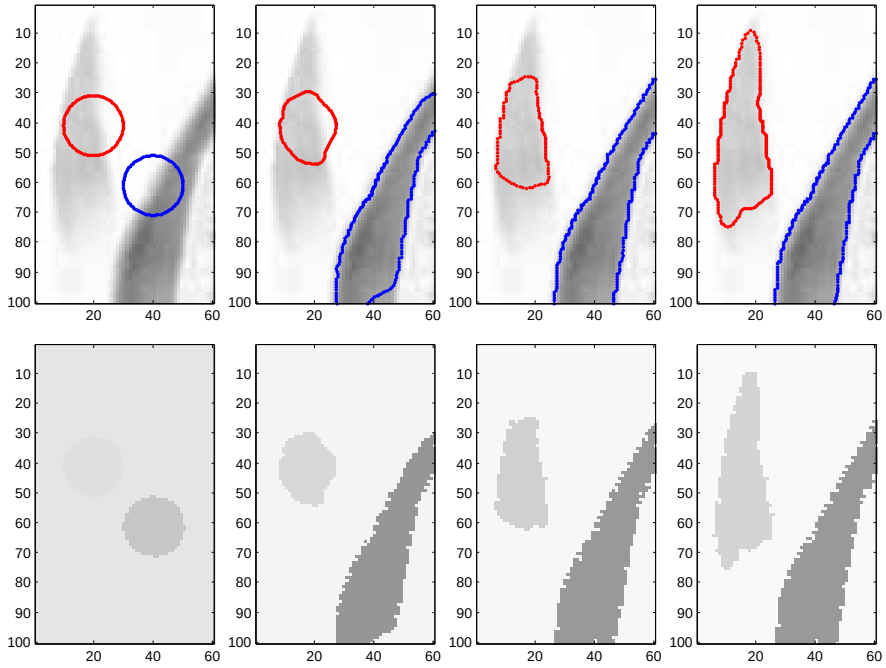


Figure 3.33: Image segmentation using Chan-Vese and piecewise constant approximation. First row: Original image and contours for $m = 1, 400, 1000, 2500$ using $\Delta t = 0.001$, $\sigma = 1$, $\lambda = 500$. Second row: Piecewise constant image approximation. Image courtesy: Dr. Declan O'Regan and the Robert Steiner MR Unit, MRC Clinical Sciences Centre, Imperial College London.

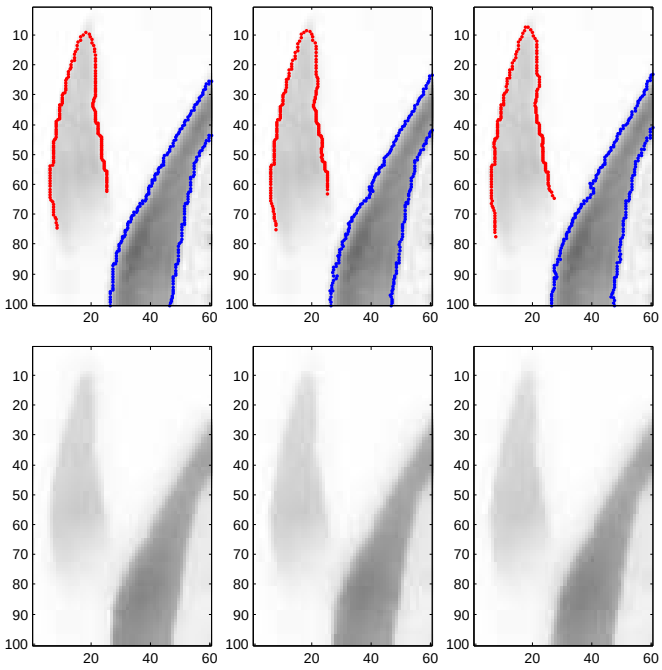


Figure 3.34: Image segmentation and denoising result. First row: Original image and contours for $m = 1, 100, 500$ using $\Delta t = 0.008$, $\sigma = 1$, $\lambda = 500$ and $\mu = 30000$. Second row: Denoised image with $\lambda = 0.1$. Image courtesy: Dr. Declan O'Regan and the Robert Steiner MR Unit, MRC Clinical Sciences Centre, Imperial College London.

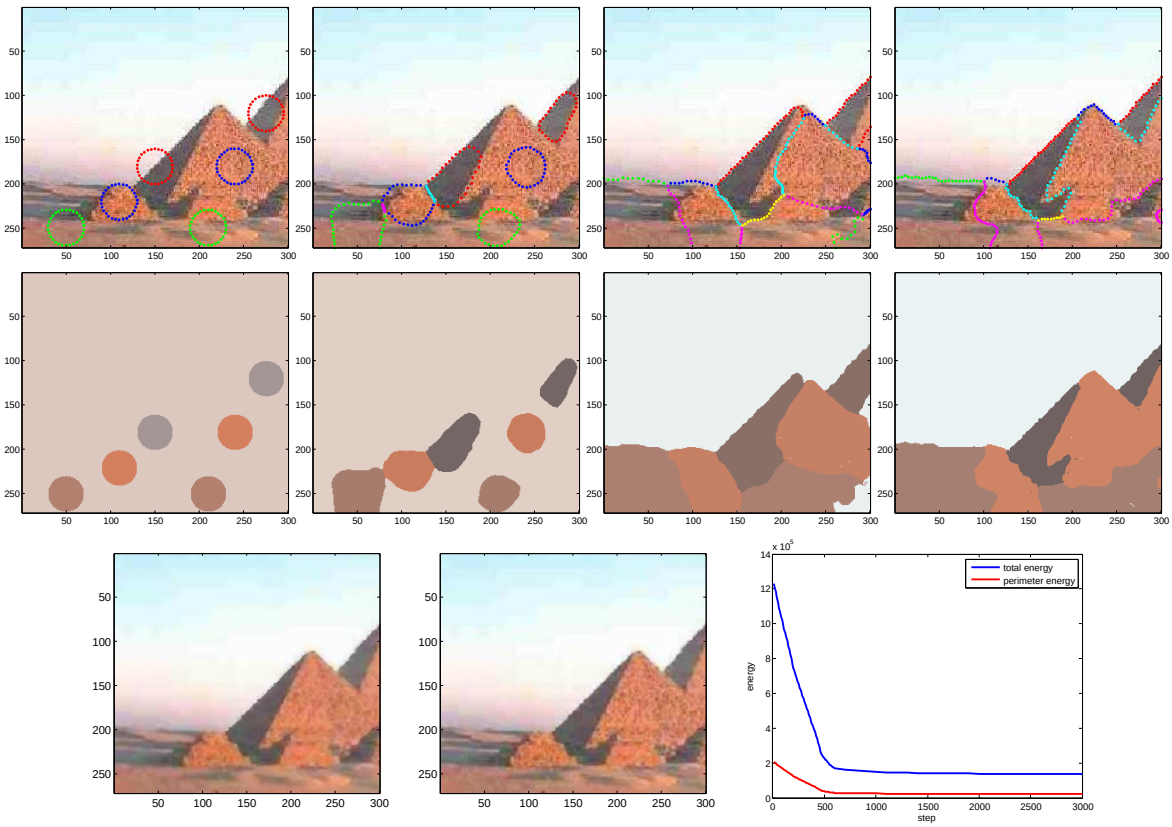


Figure 3.35: Multiphase image segmentation of a color image using CB space, with $\Delta t = 0.005$, $\lambda_C = 80$, $\lambda_B = 20$, σ-factor 20%. First row: Original image and contours for $m = 1, 100, 500, 3000$. Second row: Piecewise constant approximation. Third row: Approximation u computed in the postprocessing step with $\lambda = 1$ (initial value) and $\lambda = 5.916$ (final value after 15 iterations, portion of E_1: $\alpha = 0.2$) and plot of the Mumford-Shah energy. The original image is from the Caltech 101 dataset (Fei-Fei et al., 2004).

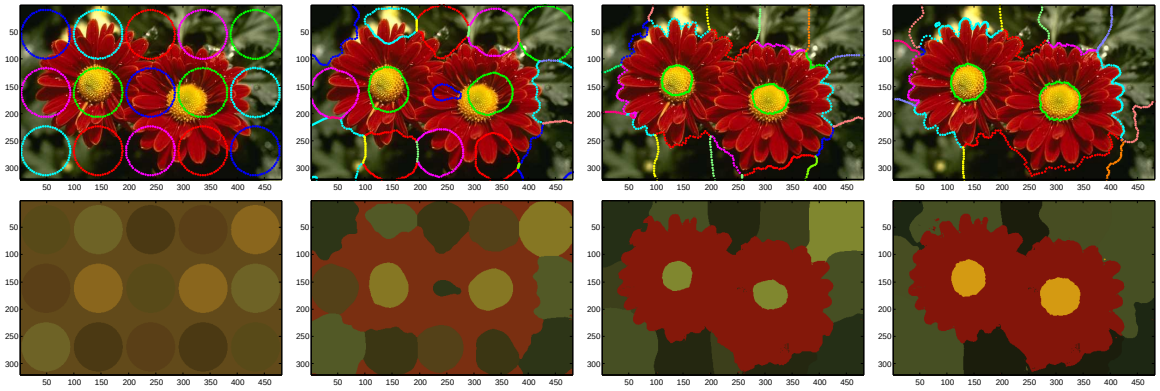


Figure 3.36: Multiphase image segmentation of a color image using HSV space, with $\Delta t = 0.01$, $\lambda_H = 150$, $\lambda_S = 50$, $\lambda_V = 50$, σ-factor 15%. First row: Original image and contours for $m = 1, 25, 100, 1900$. Second row: Piecewise constant approximation. The original image is from the Berkeley Segmentation Data Set and Benchmarks 500 (BSDS500) (Arbelaez et al., 2011; Martin et al., 2001).

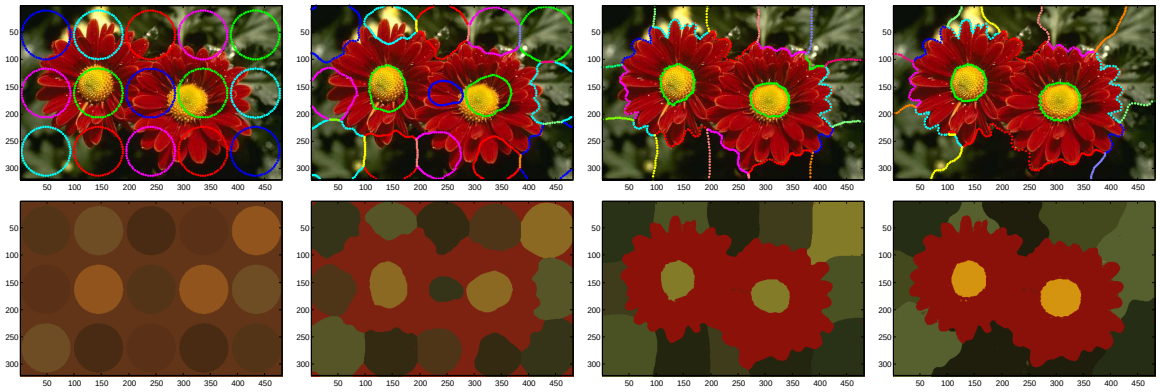


Figure 3.37: Multiphase image segmentation of a color image using CB space, with $\Delta t = 0.01$, $\lambda_C = 180$, $\lambda_B = 40$, σ-factor 15%. First row: Original image and contours for $m = 1, 50, 100, 1900$. Second row: Piecewise constant approximation. The original image is from the Berkeley Segmentation Data Set and Benchmarks 500 (BSDS500) (Arbelaez et al., 2011; Martin et al., 2001).

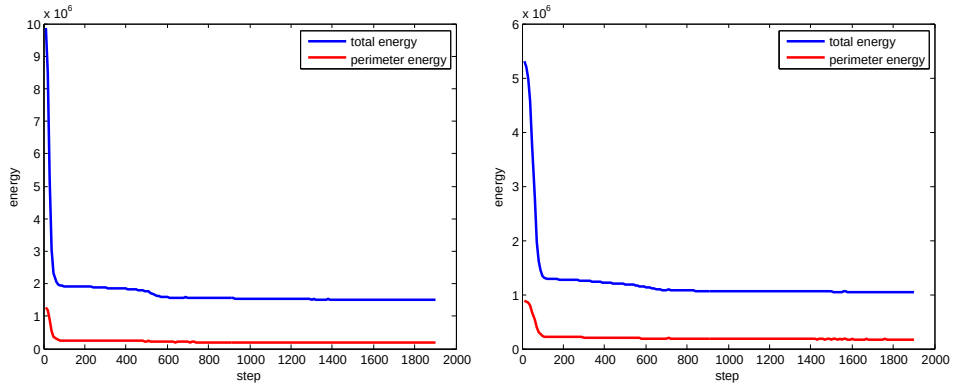


Figure 3.38: Energy decrease segmenting the flower image from the Berkeley Segmentation Data Set and Benchmarks 500 (BSDS500). Left: HSV color space. Right: CB color space.

Figure 3.39: Result of postprocessing smoothing of the flower image from the Berkeley Segmentation Data Set and Benchmarks 500 (BSDS500) (Arbelaez et al., 2011; Martin et al., 2001). First row: denoising result with initial value of $\lambda = 1$ (left: use of detected regions from HSV-segmentation, right: use of detected regions from CB-segmentation). Second row: denoising with automatic parameter setting with $\alpha = 0.2$. Left: $\lambda = 7.374$ (12 iterations, HSV). Right: $\lambda = 7.146$ (12 iterations, CB).

Figure 3.40: Image segmentation and restoration of the flower image from Berkeley Segmentation Data Set and Benchmarks 500 (BSDS500) (Arbelaez et al., 2011; Martin et al., 2001) with additional noise. First row: Image segmentation results using CB color space for $m = 6000$, $\Delta t = 0.01$, $\lambda_C = 180$, $\lambda_B = 40$, σ-factor 15%. Second row: Magnification of the noise added image (left) and denoised image (right) using $\lambda_k = 10$, $k = 1, \ldots, N_R$.

to demonstrate the algorithm and to compare the color spaces hue-saturation-value (HSV) and chromaticity-brightness (CB). Figures 3.36 and 3.37 present the results, i.e. the contours and the piecewise constant segmentation at different time steps. Apart from the weights for external forcing terms, the same parameters are used for the two runs. The segmentation using the CB color space results in a better final segmentation. The interface between the red phase and the dark green phase near the right flower matches better with the edges of the flower using the CB space. Furthermore, the different green phases of the background with different brightnesses are segmented more accurately. Figure 3.38 shows the energy decrease. In each case, the fastest energy decrease can be noted at the beginning, i.e. in the first 200 steps. Then, only small improvements of the segmentation are achieved where the energy decreases only slightly. The result of the postprocessing image denoising step u, i.e. the solution of $-\frac{1}{\lambda}\Delta u + u = u_0$ for different λ (initial value and result of automatic setting) with Neumann boundary conditions, is presented in Figure 3.39.

To demonstrate the ability to handle noisy images, noise is added to the original flower image. Figure 3.40 shows the segmentation and restoration results using the chromaticity-brightness color space. Even the noisy image can be segmented and denoised with success.

Tracking

To present another practical example, we apply our algorithm on a gray-scaled image showing a satellite with bright solar panels. The image has been generated using a camera simulator developed by Astos Solutions GmbH (2013). Figure 3.41 shows the contours during their evolution and the piecewise constant approximation. This example demonstrates again the detection and creation of triple junctions and splitting of curves. As the body consists of different gray-values (nearly white solar panels, darker body and antenna) multiphase image segmentation is applied. A practical application is camera-based relative navigation. Having detected the edges of the main body of the satellite and its solar panels, an estimation of the satellite's position and orientation relative to the camera can be made if geometry and real sizes of the spacecraft and camera parameters like focal length are known.

In this example, one inner contour disappears at the end of the evolution, i.e. the darker, rectangular object on the satellite body is assigned to the same region as the surrounding parts of the main satellite body. The reason for this is, that the Mumford-Shah energy of the final decomposition of the image in regions at $m = 1000$ is smaller compared to the decomposition at $m = 300$. With additional regions and contours (for example four-phase segmentation instead of three-phase segmentation) one could detect also details on the satellite's surface. However, for practical applications like navigation often the outer (main) contours are of interest; therefore the detection of inner contours is of minor importance.

Figure 3.42 shows the final piecewise smooth image for different smoothing parameters (initial value $\lambda = 1$ and final value $\lambda = 18.28$, portion of E_1: 20%) computed as postprocessing step. Since there is little noise in the original image, the denoised versions differ only slightly from the original image. We note that for navigation purposes image restoration is of minor interest and will rarely be performed in practice, whereas the segmentation of the image and resulting object detection is the major task.

We now demonstrate how the method can be extended to tracking applications: Figure 3.43 shows a short sequence of six images showing a rotating satellite. It presents the contours and piecewise constant approximation of the initial contours ($m = 0$) and of the final contours ($m = M = 20$).

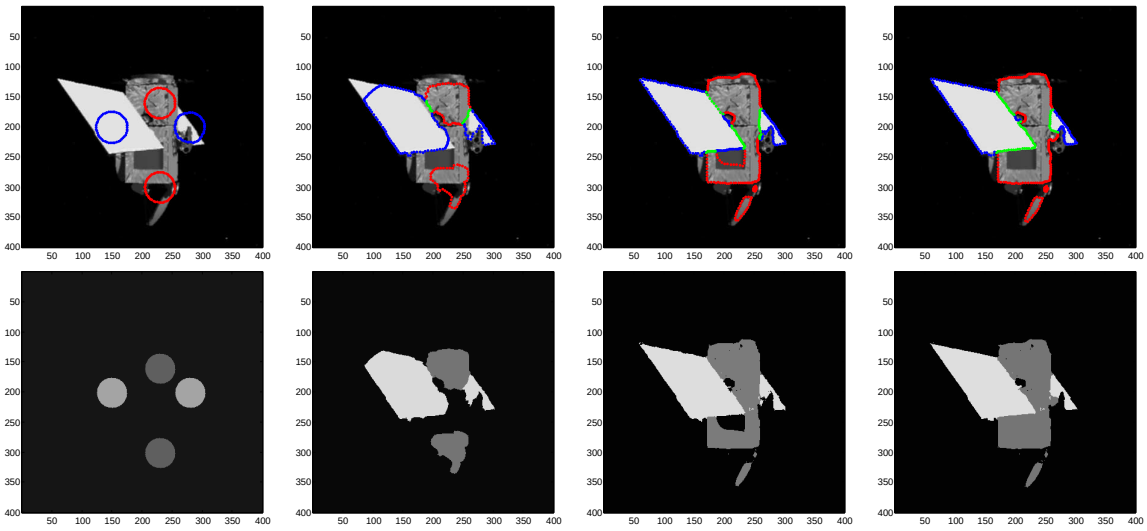


Figure 3.41: Multiphase image segmentation of a gray-scaled image showing a satellite with $\Delta t = 0.02$, $\lambda = 80$, σ-factor 25%. First row: Original image and contours for $m = 1, 50, 300, 1000$. Second row: Piecewise constant approximation.

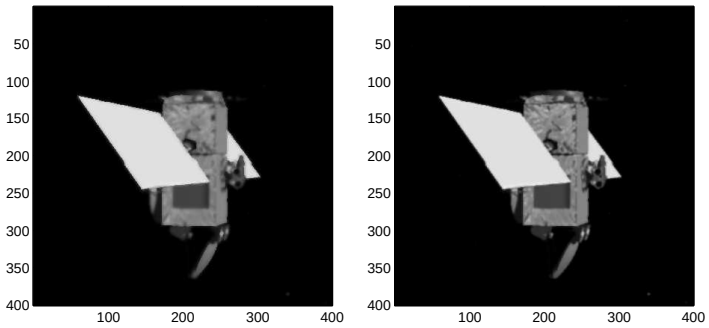


Figure 3.42: Denoised image computed in the postprocessing step using automatic setting of λ with $\alpha = 0.2$. Left: $\lambda = 1$ (initial value). Right: $\lambda = 18.280$ (after 11 iterations).

The tracking principle is simple: As proposed by Moelich and Chan (2003), we use the final curves of the previous image as initial curves for the current image. The initial curves for the first image are given by the result of the image segmentation presented in Figure 3.41. The so-called traveling distance, i.e. the necessary position change of the curves from one image to the next, is small, since successive images differ only slightly. This is why only a small number of iteration steps are needed during tracking. In our example, $M = 20$ iterations are sufficient. In general, the tracking method can fail, if the motion of the body is too fast or the image rate is too small. In this case, one can enlarge the initial contours, as discussed in Moelich and Chan (2003).

Table 3.2 presents the computational time of the MATLAB implementation of the method. In the demonstration scenario, the actual time between two successive images is 1s, i.e. the image sequence is generated with 1Hz. The MATLAB implementation is not yet fully real-time capable since the needed computational times for one image are around 1.3-1.4s. However, the time for one image during the tracking can be expected to be less than 1s and thus real-time capable with a full C/C++ implementation (currently only some routines are implemented in C and called via mexFunctions by the MATLAB main routine, cf. Section 3.4.1). For tracking, we use a fixed time step size and do not perform an adaptive setting, cf. Section 3.3.6. Further, we do not repeat a step and decrease the time step size when a topology change is executed, cf. Section 3.3.5. For real-time applications, static settings help to keep the computational time small and or at least controllable.

We do not perform an image smoothing at the end of the segmentation of the single images. As discussed above, in applications, usually only the moving body should be tracked. An image restoration need not be done in most cases. For computing the regions, we start with the piecewise constant approximation of the image and the coefficients computed in the last step for the previous image. Although the image function u_0 has changed, we update the coefficients only close to the curves as described in Section 3.3.6. For most tracking examples, the image changes only slightly and computation time can be saved if an update of the coefficients c_k, $k = 1, \ldots, N_R$, is only computed in a small band around the curves. Typically, the coefficients need not be determined with high accuracy; i.e. for a segmentation of the image, they do not need to be exactly the mean of the gray value in the corresponding region. The coefficients are only needed for the external energy. Therefore, rough values of the coefficients c_k are sufficient such that the curves evolve in the correct direction. However, if a body is tracked over a longer sequence of images where the visible surfaces of the body, the lightning or the background changes, one need to recompute the average of the image function in the regions, by evaluating the image function in the entire 2D image domain, not only in a small band around the curves. Such an update depends on how fast the scene changes and is usually determined by the application. In case of satellite optical navigation, the change of the scene and thus the need to update the coefficients in the entire 2D image domain typically depends on the orbit and on the angle between observer (camera), object and the Sun.

The tracking example also demonstrates that topology changes can occur when a moving body is tracked. We have chosen a scenario in which a small part of one solar panel is outside the field of view of the camera in the last three images. Thus, an intersection with the image boundary occurs. The third image shows a situation where the blue curve nearly touches the image boundary (third row in Figure 3.43). When segmenting the fourth image, the curve intersects the image boundary and the curve is split into two curves. Another topology change takes place involving the two small red curves enclosing an antenna of the satellite which merge to one single curve.

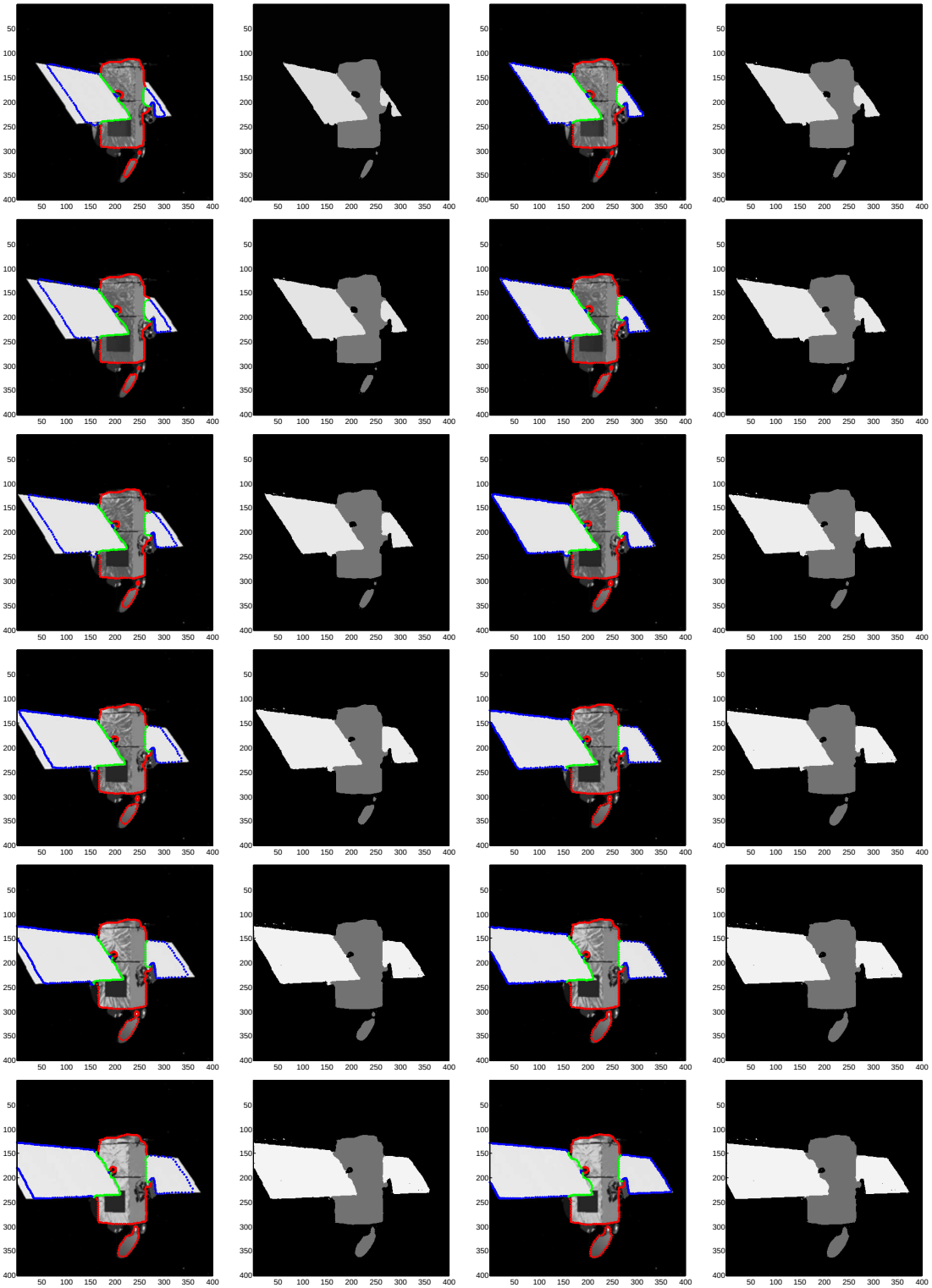


Figure 3.43: Tracking a satellite over a sequence of 6 images with $\Delta t = 0.02$, $\lambda = 80$, $\sigma = 50$. First column: Original image and contours for $m = 1$. Second column: Piecewise constant approximation for $m = 1$. Third column: Original image and contours for $m = M = 20$. Fourth column: Piecewise constant approximation for $m = M = 20$.

Image Nr	Number of Nodes	Computational Time [s]
1	359	1.4173
2	370	1.2995
3	382	1.3000
4	400	1.3483
5	408	1.3671
6	435	1.4171

Table 3.2: Computational time during tracking, number of nodes and measured time for segmentation of one image of the sequence (2.4GHz CPU, MATLAB implementation, usage of a single core).

3.A Complex planar curve network situations

In this Appendix we discuss how the curve evolution method and the topology change algorithm can handle complex situations. We consider some exemplary cases which can occur sporadically during the evolution of a curve network. Although the discussed cases are quite complex they all can be dealt with by the basic algorithm stated in Section 3.3.5.

First, we consider the case in which a boundary node approaches a corner of the rectangular image domain. We distinguish two cases illustrated in Figure 3.44 and 3.45. In the situation shown in Figure 3.44, the boundary node $\vec{Q}_k \subset \partial\Omega$ approaches a corner. Its direct neighbor node is also in a boundary square of the auxiliary grid used by the algorithm for topology changes, cf. Section 3.3.5. Thus, a boundary contact is detected for the neighbor node. The node is projected to the boundary and duplicated, resulting in two new boundary nodes $\vec{Q}_{k_2}$ and $\vec{Q}_{k_3}$, and the curve splits up in two separate curves. The two duplicated nodes are drawn in Figure 3.44 with a small distance for illustration reasons. In real, their position is identical. Finally, the curve connecting $\vec{Q}_k$ and $\vec{Q}_{k_2}$ is deleted since its length is too small. We assume that the minimal allowed length for a curve, such that it is not deleted, is larger than the distance between neighbor nodes and larger than the grid size of the auxiliary grid.

Another situation is presented by Figure 3.45, where a boundary node $\vec{Q}_k$ moves along the boundary of the image domain. In one time step, drawn in the first subfigure, the distance between the node and the corner is larger than the grid size a. No boundary intersection is thus detected for its neighbor. In the next step, the boundary node moves in positive x-direction and exceeds the image domain. All nodes which are outside of the image domain are reprojected to the boundary of the image domain. We now change the outer normal vector at $\vec{Q}_k$ from $(0,1)^T$ to $(1,0)^T$ according to the following considerations: The boundary of the rectangular image domain can be divided in four parts with four different normal vectors. Each boundary node has to be assigned to one of the four boundary parts, such that we can define an outer normal at $\vec{Q}_k$. At corners, two boundary parts intersect and a unique assignment is not possible. If a node approaches the corner but does not exceed the image domain, we keep the previous assignment and thus keep the direction of the outer normal. However, if a node exceeds the image domain, we change the assignment. We reproject $\vec{Q}_k$ back to the corner, and assign the node to the other possible boundary part.

A boundary intersection is then detected for the neighbor node of $\vec{Q}_k$. Again, the new boundary node is duplicated, the curve splits up in two intermediate curves, where the smaller curve is deleted. Some other variants can also occur, where further neighbor nodes lie inside of boundary squares and or exceed the image domain. Such case are handled as described

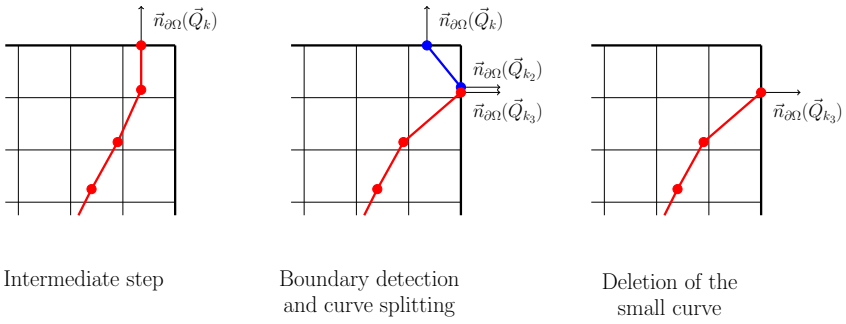


Figure 3.44: Approaching a corner (case 1).

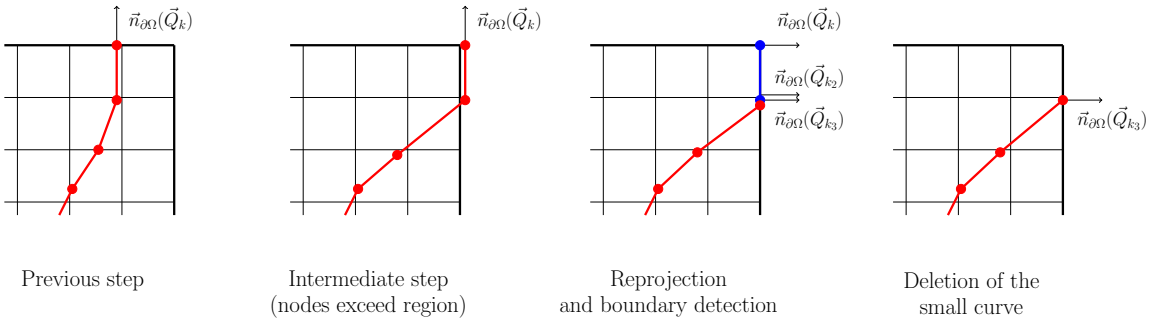
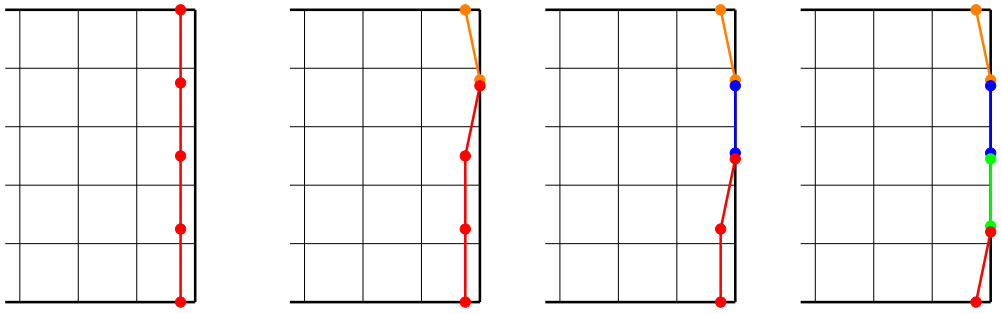


Figure 3.45: Approaching/exceeding a corner (case 2).

above: nodes are projected to the boundary, the curve is split into two or more curves followed by a deletion of the small curves.

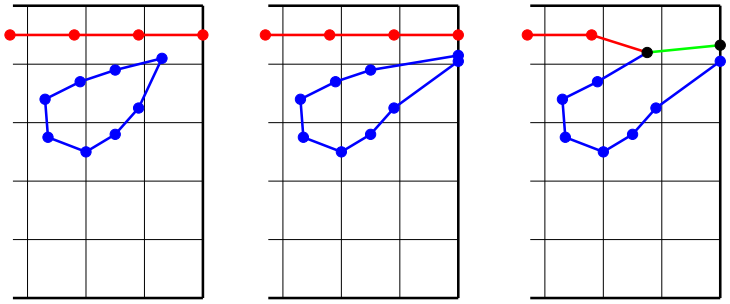
A similar situation occurs if several node points on a straight line approach the boundary of the image domain at the same time. Figure 3.46 presents a line with 5 nodes. The three inner nodes are detected as new boundary intersection nodes. The nodes which belong to $\partial\Gamma \subset \partial\Omega$ have already been boundary nodes. The three new nodes are collected in a list for boundary intersection points and are now considered one by one, see subfigure 2-4: Each node is projected to the boundary and duplicated. The curve to which the node has belonged to is split up. In the example illustrated in Figure 3.46 this procedure results in four small, single curves, each consisting of two nodes. We can assume that the minimum allowed length for a curve is larger than the distance between two neighbor nodes. Thus, all intermediately created curves are directly deleted due to their small length.

Another interesting situation occurs, when a curve meets with the image boundary at a location close to the endpoint of another curve, see Figure 3.47. In such a situation, a boundary intersection and a triple junction occurs at the same location. This complex situation can be handled by applying the single methods for creation of boundary nodes and triple junctions one by one: First, the node is projected to the image boundary, duplicated and the closed curve becomes an open curve. Then, a triple junction is created. The neighbor nodes of the two boundary points are considered to decide which boundary node of the blue curve in Figure 3.47 should be used for the creation of the new curve. Due to the performed node manipulations, the nodes (for example of the blue curve) are no longer (nearly) equidistributed; however, they will be redistributed automatically in tangential direction in the next time steps of the evolution.



Intermediate step, all nodes detected as boundary nodes | 2 intermediate curves, | 3 intermediate curves | 4 intermediate curves

Figure 3.46: A straight line approaches the boundary of the image domain.



Boundary and triple junction detection | Projection to boundary | Creation of junction and one new curve

Figure 3.47: Boundary intersection and triple junction detection at the same time.

In a situation where a boundary detection and a triple junction occurs at the same time, also neighbor nodes can be involved in topology changes, especially for convex curves. For example, if a neighbor node is also detected as a boundary intersection node, the nodes are projected one by one to the boundary of the image domain, similar as described above for the straight line. Small curves, which are immediately deleted, can emerge when more than one boundary point is detected. Finally, the triple junction creation is executed.

Chapter 4

Processing of Images on Surfaces

In this chapter we extend the methods for image segmentation and denoising developed for planar images to images defined on (curved) surfaces. For segmentation, we consider the motion of curves on surfaces which is given by mean-curvature flow related evolution laws. A parametric approach for curve evolution and an extension of the Chan-Vese method is used to segment images on surfaces. The analytical and numerical framework for image segmentation with parametric curve evolution on surfaces is described in this chapter. The method to detect topology changes is extended from planar curves to curves on surfaces. For denoising with edge enhancement, a diffusion equation with Neumann boundary conditions is solved on the surface. We conclude this chapter by presenting results of image segmentation and restoration applied on artificial and real images.

4.1 Introduction

Let $\mathcal{M} \subset \mathbb{R}^3$ be a given surface, i.e. a two-dimensional manifold in $\mathbb{R}^3$. An image defined on the surface is given by an image function $u_0 : \mathcal{M} \to [0,1]^d$. Each point on the surface is assigned to a gray value ($d = 1$) or to a color (e.g. $d = 3$).

For image segmentation with active contours, we consider an evolving curve $\Gamma(t) \subset \mathcal{M}$, $t \in [0, T]$. Evolution laws designed for image segmentation describe the motion of the curve. As in the two-dimensional case, the evolution laws, which we consider in this chapter, contain a curvature term which provides smoothness of the curve and an external forcing term which pushes the contour to the edges of objects or to region boundaries. Evolution laws and methods like the Chan-Vese model can be extended from the planar case to images which are defined on surfaces. Since $\Gamma(t)$ evolves on a surface, an additional condition is needed which forces the curve to stay on the manifold $\mathcal{M}$. This is why the contour $\Gamma(t)$ cannot be handled as just a curve in $\mathbb{R}^3$. For the curvature term, we need to make use of the *geodesic curvature*, recall Section 2.2.

The method for topology changes developed for the planar case is extended to curves on two-dimensional manifolds in $\mathbb{R}^3$. Using the Euclidean distance in $\mathbb{R}^3$ to detect topology changes can lead to false detections: Surfaces exist where two points can have a small Euclidean distance but their *geodesic distance* is large. Figure 4.1 illustrates an example where the use of the Euclidean distance would lead to false detections. In such situations, a topology change does not occur. The geodesic distance would be a better indicator for proximity detection compared to the Euclidean distance. However, the computation of the

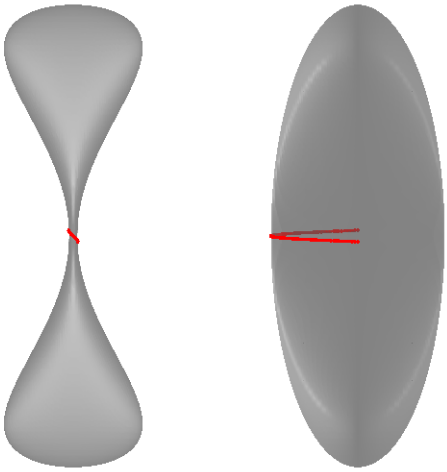


Figure 4.1: Example of a surface observed from two different viewing angles (front and side view) and plot of an exemplary curve on the surface. The endpoints of the curve have a small Euclidean distance but a large geodesic distance.

geodesic distance between two points on a surface is very expensive from a computational point of view and cannot be used in a sub-algorithm in practice. We propose a solution to the problem of detecting topology changes in Section 4.3.4.

Besides topology changes, more practical problems arise when considering real image data. Real surfaces are typically given as a discrete data set, often as triangulated surfaces. The image data is given piecewise for each vertex or each triangle. The evaluation of the image function at a point $\vec{x}$ of the curve or the computation of a normal to the surface $\mathcal{M}$ at $\vec{x}$ is not directly possible knowing only the coordinates of $\vec{x}$. The point $\vec{x}$ has to be first assigned to a triangle of the triangulation. An efficient assignment of points $\vec{x}$ to surface triangles is presented in Section 4.3.5.

Further, for solving a diffusion scheme like (3.43), we need to solve surface partial differential equations. For example, the Laplace operator has to be replaced by its analogue for surfaces, the Laplace-Beltrami operator. We derive and discuss the surface partial differential equations and the resulting image restoration scheme in Section 4.2.5.

To sum up, the extension of the method developed for planar images leads to several conceptional and practical challenges when considering image segmentation and image denoising on surfaces.

4.2 Methods for Image Segmentation and Restoration

In this chapter, $\mathcal{M} \subset \mathbb{R}^3$ is a smooth, two-dimensional manifold which can be represented as zero level set of a smooth function $\Phi \in C^1(\mathbb{R}^3, \mathbb{R})$. As in Section 2.2, let $\vec{n}_\Phi = \nabla\Phi/\|\nabla\Phi\|$ be a normal to the surface and for a curve $\Gamma \subset \mathcal{M}$ parameterized by $\vec{x} : I \to \mathcal{M}$, we set $\vec{\nu}_\Phi := \vec{n}_\Phi \circ \vec{x}$ and $\vec{\nu}_\mathcal{M} = \vec{x}_s \times \vec{\nu}_\Phi$. Thus, $\vec{\nu}_\mathcal{M}(\rho)$ is a vector which is normal to the curve Γ but tangent to the surface $\mathcal{M}$ for each $\rho \in I$, where I is a one-dimensional reference manifold, e.g. $I = [0, 1]$ for open curves and $I = \mathbb{R}/\mathbb{Z}$ for closed curves.

4.2.1 Edge-based Active Contours on Surfaces

In Section 3.2.3, we discussed the advantages of region-based active contours methods compared to edge-based methods. This is why we will restrict our considerations to region-based methods in the numerical and result section of this chapter. However, we shortly state the evolutions equations arising from the geodesic active contours model of Caselles et al. (1997a) (cf. Section 3.2.2) extended to images on surfaces.

For edge detection in the plane with the geodesic active contours model, we considered a weighted length functional (3.8) and derived the evolution law $V_n = f\kappa - \nabla f \,.\, \vec{\nu}$, where f was an edge indicator function defined on the image domain. In detail, the positive function f was small at locations where the gradient of the image function was big, and f was larger in homogeneous parts of the image. We now consider an analogue energy to (3.8) for curves on surfaces and derive corresponding evolution equations.

Therefore, let $u_0 : \mathcal{M} \to [0,1]$ be a given image function and $u : \mathcal{M} \to [0,1]$ an approximation of u_0. For example, u can be the result of a preceding image smoothing. Let $f : \mathcal{M} \to \mathbb{R}$ be a smooth, positive edge indicator function which is small at edges (i.e. at locations where u changes rapidly) and larger in homogeneous regions (i.e. at locations where u is nearly constant).

Let $\Gamma \subset \mathcal{M}$ be a closed curve on the surface. We consider the energy

$$E(\Gamma) = \int_\Gamma f \,\mathrm{d}s = \int_I f(\vec{x}) \|\vec{x}_\rho\| \,\mathrm{d}\rho, \tag{4.1}$$

where $\vec{x} : I \to \mathcal{M}$ is a parameterization of Γ.

We now derive flows for image segmentation as gradient flows using methods of the theory of calculus of variations. The variations are restricted to lie on the surface $\mathcal{M}$. Therefore, we introduce the space

$$\underline{V}_\Phi = \left\{ \vec{\eta} : I \to \mathbb{R}^3 \,:\, \vec{\eta} \text{ is smooth and } \vec{\eta} \,.\, \vec{\nu}_\phi = 0 \right\} \tag{4.2}$$

and define for functions $\vec{\eta}, \vec{\chi} : I \to \mathbb{R}^3$ the following inner product

$$(\vec{\eta}, \vec{\chi})_{2,\mathcal{M},nor} := \int_\Gamma \vec{P}_\mathcal{M} \vec{\eta} \,.\, \vec{P}_\mathcal{M} \vec{\chi} \,\mathrm{d}s, \tag{4.3}$$

where $\vec{P}_\mathcal{M}$ is the projection onto the part in direction $\vec{\nu}_\mathcal{M}$, i.e. $\vec{P}_\mathcal{M}\vec{\eta} = (\vec{\eta} \,.\, \vec{\nu}_\mathcal{M})\vec{\nu}_\mathcal{M}$ for $\vec{\eta} : I \to \mathbb{R}^3$.

For $\vec{\eta} \in \underline{V}_\Phi$, we consider $\vec{y} : I \times (-\epsilon_0, \epsilon_0) \to \mathcal{M}$ with $\vec{y}(\rho, 0) = \vec{x}(\rho)$ and $\vec{y}_\epsilon(\rho, 0) = \vec{\eta}(\rho)$.

Let $\Gamma^\epsilon \subset \mathcal{M}$ be the image of $\vec{y}(\,.\,, \epsilon)$ and let $\vec{\nu}^\epsilon_\mathcal{M}(\rho) \in T_{\vec{y}(\rho,\epsilon)}\mathcal{M}$ be the vector in the tangent space to $\mathcal{M}$ in $\vec{y}(\rho, \epsilon)$ defined by $\vec{\nu}^\epsilon_\mathcal{M}(\rho) = \vec{y}_s(\rho, \epsilon) \times \vec{n}_\Phi(\vec{y}(\rho, \epsilon))$. We define

$$(\delta E(\Gamma))(\vec{\eta}) := \left.\frac{\mathrm{d}}{\mathrm{d}\epsilon}\right|_{\epsilon=0} E(\Gamma^\epsilon) = \left.\frac{\mathrm{d}}{\mathrm{d}\epsilon}\right|_{\epsilon=0} \int_I f(\vec{y}) \|\vec{y}_\rho\| \,\mathrm{d}\rho,$$

on noting that $\vec{y}(\,.\,,\epsilon)$ and Γ^ϵ depend on $\vec{\eta}$. We compute

$$\begin{aligned}
(\delta E(\Gamma))(\vec{\eta}) &= \frac{\mathrm{d}}{\mathrm{d}\epsilon}\bigg|_{\epsilon=0} \int_I f(\vec{y})\|\vec{y}_\rho\|\,\mathrm{d}\rho \\
&= \left(\int_I (\nabla_{\mathcal{M}} f(\vec{y})\,.\,\vec{y}_\epsilon\|\vec{y}_\rho\| + f(\vec{y})\frac{\vec{y}_\rho}{\|\vec{y}_\rho\|}\,.\,\vec{y}_{\rho\epsilon})\,\mathrm{d}\rho\right)\bigg|_{\epsilon=0} \\
&= \int_I (\nabla_{\mathcal{M}} f(\vec{x})\,.\,\vec{\eta}\,\|\vec{x}_\rho\| + f(\vec{x})\frac{\vec{x}_\rho}{\|\vec{x}_\rho\|}\,.\,\vec{\eta}_\rho)\,\mathrm{d}\rho \\
&= \int_\Gamma (\nabla_{\mathcal{M}} f(\vec{x})\,.\,\vec{\eta} + f(\vec{x})\,\vec{x}_s\,.\,\vec{\eta}_s)\,\mathrm{d}s \\
&= \int_\Gamma (\nabla_{\mathcal{M}} f(\vec{x}) - (\nabla_{\mathcal{M}} f(\vec{x})\,.\,\vec{x}_s)\vec{x}_s - f(\vec{x})\vec{x}_{ss})\,.\,\vec{\eta}\,\mathrm{d}s \\
&= \int_\Gamma ((\nabla_{\mathcal{M}} f(\vec{x})\,.\,\vec{\nu}_{\mathcal{M}})\vec{\nu}_{\mathcal{M}} - f(\vec{x})\kappa_{\mathcal{M}}\vec{\nu}_{\mathcal{M}})\,.\,\vec{\eta}\,\mathrm{d}s. \qquad (4.4)
\end{aligned}$$

For the first identity we used the chain rule (2.47) and for the second last identity integration by parts. For the last identity we used that $\vec{x}_s(\rho)$ and $\vec{\nu}_{\mathcal{M}}(\rho)$ is a basis of $T_{\vec{x}(\rho)}\mathcal{M}$ and (2.46) together with $\vec{\eta} \in \underline{V}_\Phi$.

A time-dependent function $\vec{x} : I \times [0,T] \to \mathcal{M}$ is called a solution to the gradient flow equation if for all $\vec{\eta} \in \underline{V}_\Phi$

$$(\vec{x}_t, \vec{\eta})_{2,\mathcal{M},nor} = -\left(\delta E(\Gamma)\right)(\vec{\eta}) \qquad (4.5)$$

holds. Further, the condition $\vec{x}_t(\,.\,,t) \in \underline{V}_\Phi$ needs to hold such that the curve stays on the surface $\mathcal{M}$, or, in other words, such that $\vec{x}(\,.\,,t)$ is a mapping onto $\mathcal{M}$.

Let $\vec{\eta} \in \underline{V}_\Phi$, we conclude from (4.4) and (4.5)

$$\vec{P}_{\mathcal{M}}\vec{x}_t = f\,\kappa_{\mathcal{M}}\vec{\nu}_{\mathcal{M}} - (\nabla_{\mathcal{M}} f\,.\,\vec{\nu}_{\mathcal{M}})\vec{\nu}_{\mathcal{M}}. \qquad (4.6)$$

Note that $\vec{\nu}_{\mathcal{M}}\,.\,\vec{\eta} = \vec{\nu}_{\mathcal{M}}\,.\,\vec{P}_{\mathcal{M}}\vec{\eta}$.

Let $V_n := \vec{x}_t\,.\,\vec{\nu}_{\mathcal{M}}$ denote the *normal velocity*, i.e. the velocity in direction $\vec{\nu}_{\mathcal{M}}$. Then $\vec{P}_{\mathcal{M}}\vec{x}_t = V_n\vec{\nu}_{\mathcal{M}}$ and consequently the equation above can be rewritten to

$$V_n = f\kappa_{\mathcal{M}} - \nabla_{\mathcal{M}} f\,.\,\vec{\nu}_{\mathcal{M}}. \qquad (4.7)$$

This equation is the analogue to (3.13), where the curvature κ is replaced by the geodesic curvature $\kappa_{\mathcal{M}}$, the gradient ∇f by the surface gradient $\nabla_{\mathcal{M}} f$ and the normal $\vec{\nu}$ by the normal $\vec{\nu}_{\mathcal{M}}$ which is normal to the curve but lies in the tangent space to $\mathcal{M}$.

Remark 4.1. *The geodesic active contours model aims at minimizing the energy*

$$E(\Gamma) = \int_\Gamma f\,\mathrm{d}s = \int_I f(\vec{x}(\rho))\|\vec{x}_\rho(\rho)\|\,\mathrm{d}\rho.$$

Let $\vec{\tau}_1, \vec{\tau}_2$ be smooth vector fields such that $\{\vec{\tau}_1(\vec{p}), \vec{\tau}_2(\vec{p})\}$ is an orthonormal basis of $T_{\vec{p}}\mathcal{M}$ for each $\vec{p} \in \mathcal{M}$.

For the positive, smooth edge indicator function f we define $g_{ij} = (f\vec{\tau}_i)\,.\,(f\vec{\tau}_j)$, $i,j = 1,2$. This defines a family $g = (g_{\vec{p}})_{\vec{p}\in\mathcal{M}}$ of inner products $g_{\vec{p}} : T_{\vec{p}}\mathcal{M} \times T_{\vec{p}}\mathcal{M} \to \mathbb{R}$ with

$$g_{\vec{p}}(\vec{u}_{\vec{p}}, \vec{v}_{\vec{p}}) := \sum_{i,j=1}^{2} g_{ij}(\vec{p})(\vec{u}_{\vec{p}}\,.\,\vec{\tau}_i(\vec{p}))(\vec{v}_{\vec{p}}\,.\,\vec{\tau}_j(\vec{p})),$$

for $\vec{u}_{\vec{p}}, \vec{v}_{\vec{p}} \in T_{\vec{p}}\mathcal{M}$. Further the mapping

$$\vec{p} \mapsto g_{\vec{p}}(\vec{X}(\vec{p}), \vec{Y}(\vec{p}))$$

is differentiable with respect to $\vec{p}$ for all differentiable tangential vector fields $\vec{X}, \vec{Y}$. Therefore, g defines a Riemannian metric and $(\mathcal{M}, g)$ is a Riemannian manifold. The energy $E(\Gamma)$ can be expressed as

$$E(\Gamma) = \int_I \sqrt{\sum_{i,j=1}^{2} g_{ij}(\vec{x}_\rho \,.\, \vec{\tau}_i)(\vec{x}_\rho \,.\, \vec{\tau}_j)}\, \mathrm{d}\rho.$$

Now we consider an open curve Γ on $\mathcal{M}$ with endpoints $\vec{x}_0 := \vec{x}(0)$ and $\vec{x}_1 := \vec{x}(1)$. $E(\Gamma)$ is the length of the curve Γ with respect to the metric g. If $\vec{x}$ minimizes the curve length and is parameterized by arc-length with respect to the metric g, then the curve $\Gamma = \vec{x}(I)$ is a geodesic.

In summary, a minimum of the functional $E(\Gamma)$ is the shortest path between $\vec{x}_0$ and $\vec{x}_1$ with respect to the metric g. For $f \equiv 1$, the metric g reduces to the standard metric given by $g_{ij} = \vec{\tau}_i \,.\, \vec{\tau}_j$ and the flow (4.7) reduces to the geodesic curvature flow $V_n = \kappa_{\mathcal{M}}$.

4.2.2 The Mumford-Shah Functional and Region-based Active Contours on Surfaces

We consider an extension of the Mumford-Shah and Chan-Vese model to images of surfaces: Let $u_0 : \mathcal{M} \to [0, 1]$ be a given image function. We consider the following functional:

$$E^{\mathrm{MS}}(u, \Gamma) = \sigma|\Gamma| + \int_{\mathcal{M}\setminus\Gamma} \|\nabla_{\mathcal{M}} u\|^2 \,\mathrm{d}A + \lambda \int_{\mathcal{M}} (u_0 - u)^2 \,\mathrm{d}A, \tag{4.8}$$

where $\mathrm{d}A$ is the area element (recall Section 2.1.1), $\Gamma \subset \mathcal{M}$ is a union of curves (each curve with no self-intersections) and $|\Gamma|$ denotes the length of the curves (cf. (2.50)). Further, $u : \mathcal{M} \to \mathbb{R}$ is a function which serves as an approximation of u_0, and $\sigma, \lambda > 0$ are weighting parameters.

Compared to the planar Mumford-Shah functional (3.16), the gradient ∇u is replaced by the surface gradient $\nabla_{\mathcal{M}} u$.

As in the Section 3.2.3, we consider the case of interface curves which each separate two regions, i.e. open subsets of $\mathcal{M}$, and a piecewise constant approximation u of u_0.

Before introducing a multiphase formulation, we consider the case of one closed curve Γ on $\mathcal{M}$ which is homotopic to a point. Then, the curve divides $\mathcal{M}$ in two disjoint regions Ω_1 and Ω_2 such that

$$\mathcal{M} = \Omega_1 \cup \Gamma \cup \Omega_2.$$

The indices of the regions Ω_k, $k = 1, 2$, are chosen such that $\vec{\nu}_{\mathcal{M}}$ points from Ω_2 to Ω_1. We consider a piecewise constant approximation with $u_{|\Omega_k} = c_k \in \mathbb{R}$, $k = 1, 2$.

The Mumford-Shah functional (4.8) reduces to

$$E(\Gamma, c_1, c_2) = \sigma|\Gamma| + \lambda \int_{\Omega_1} (u_0 - c_1)^2 \,\mathrm{d}A + \lambda \int_{\Omega_2} (u_0 - c_2)^2 \,\mathrm{d}A. \tag{4.9}$$

Similar to this functional, we can consider the analogue of the planar Chan-Vese model (3.21) for images on surfaces:

$$E(\Gamma, c_1, c_2) = \sigma \int_\Gamma 1 \,\mathrm{d}s + \mu \int_{\Omega_1} 1 \,\mathrm{d}A + \lambda_1 \int_{\Omega_1} f_1 \,\mathrm{d}A + \lambda_2 \int_{\Omega_2} f_2 \,\mathrm{d}A, \tag{4.10}$$

where $\sigma, \lambda_1, \lambda_2 > 0$, $\mu \geq 0$ are weighting parameters. Similar to the planar Chan and Vese (2001) method, the function f_k, $k = 1, 2$, is defined by

$$f_k(\vec{z}) = (u_0(\vec{z}) - c_k)^2, \quad \vec{z} \in \overline{\Omega_k} \subset \mathcal{M}. \tag{4.11}$$

The Chan-Vese model (4.10) with $\mu = 0$ and $\lambda_1 = \lambda_2$ is a special case of the Mumford-Shah model where the curves are interfaces and separate regions, and u is piecewise constant on each region.

Fixing now the curve Γ, we obtain the following condition for the coefficients c_k, $k = 1, 2$:

$$c_k = \frac{\int_{\Omega_k} u_0 \,\mathrm{d}A}{\int_{\Omega_k} 1 \,\mathrm{d}A}. \tag{4.12}$$

Let $\vec{x} : I = \mathbb{R}/\mathbb{Z} \to \mathcal{M}$ be a smooth parameterization of Γ.

We now consider c_1, c_2 as fixed. For $\vec{\eta} \in \underline{V}_\Phi$, we consider a variation $\vec{y} : I \times (-\epsilon_0, \epsilon_0) \to \mathcal{M}$ with $\vec{y}(\rho, 0) = \vec{x}(\rho)$ and $\vec{y}_\epsilon(\rho, 0) = \vec{\eta}(\rho)$.

Let $\Gamma^\epsilon \subset \mathcal{M}$ be the image of $\vec{y}(\,.\,, \epsilon)$, let $\Omega_1^\epsilon, \Omega_2^\epsilon \subset \mathcal{M}$ be regions such that $\mathcal{M} = \Omega_1^\epsilon \cup \Gamma^\epsilon \cup \Omega_2^\epsilon$.

Further, let $\vec{\nu}_{\mathcal{M}}^\epsilon(\rho) \in T_{\vec{y}(\rho,\epsilon)}\mathcal{M}$ be a vector in the tangent space to $\mathcal{M}$ in $\vec{y}(\rho, \epsilon)$ defined by $\vec{\nu}_{\mathcal{M}}^\epsilon(\rho) = \vec{y}_s(\rho, \epsilon) \times \vec{n}_\Phi(\vec{y}(\rho, \epsilon))$. The vector field $\vec{\nu}_{\mathcal{M}}^\epsilon(\rho)$ points in the direction Ω_1^ϵ and defines a normal field $\vec{\nu}_{\mathcal{M}}^\epsilon$ on Γ^ϵ.

We define

$$(\delta E(\Gamma))(\vec{\eta}) := \left.\frac{\mathrm{d}}{\mathrm{d}\epsilon}\right|_{\epsilon=0} \left(\sigma \int_{\Gamma^\epsilon} 1 \,\mathrm{d}s + \mu \int_{\Omega_1^\epsilon} 1 \,\mathrm{d}A + \lambda_1 \int_{\Omega_1^\epsilon} f_1 \,\mathrm{d}A + \lambda_2 \int_{\Omega_2^\epsilon} f_2 \,\mathrm{d}A \right)$$

on noting that $\vec{y}(\,.\,, \epsilon)$ and thus Γ^ϵ, Ω_1^ϵ and Ω_2^ϵ depend on $\vec{\eta}$. We compute

$$\begin{aligned}
(\delta E(\Gamma))(\vec{\eta}) &= \left.\frac{\mathrm{d}}{\mathrm{d}\epsilon}\right|_{\epsilon=0} \left(\sigma \int_I \|\vec{y}_\rho\| \,\mathrm{d}\rho + \mu \int_{\Omega_1^\epsilon} 1 \,\mathrm{d}A + \lambda_1 \int_{\Omega_1^\epsilon} f_1 \,\mathrm{d}A + \lambda_2 \int_{\Omega_2^\epsilon} f_2 \,\mathrm{d}A \right) \\
&= \left(\sigma \int_I \frac{\vec{y}_\rho}{\|\vec{y}_\rho\|} \,.\, \vec{y}_{\rho\epsilon} \,\mathrm{d}\rho + \mu \int_I (-\vec{y}_\epsilon \,.\, \vec{\nu}_{\mathcal{M}}^\epsilon) \|\vec{y}_\rho\| \,\mathrm{d}\rho + \right. \\
&\quad \left. + \lambda_1 \int_I f_1(\vec{y}) (-\vec{y}_\epsilon \,.\, \vec{\nu}_{\mathcal{M}}^\epsilon) \|\vec{y}_\rho\| \,\mathrm{d}\rho + \lambda_2 \int_I f_2(\vec{y}) (\vec{y}_\epsilon \,.\, \vec{\nu}_{\mathcal{M}}^\epsilon) \|\vec{y}_\rho\| \,\mathrm{d}\rho \right) \Bigg|_{\epsilon=0} \\
&= \sigma \int_\Gamma \vec{x}_s \,.\, \vec{\eta}_s \,\mathrm{d}s + \int_\Gamma (-\mu - \lambda_1 f_1 + \lambda_2 f_2) \, \vec{\nu}_{\mathcal{M}} \,.\, \vec{\eta} \,\mathrm{d}s \\
&= \int_\Gamma (-\sigma \vec{x}_{ss} + (-\mu - \lambda_1 f_1 + \lambda_2 f_2) \, \vec{\nu}_{\mathcal{M}}) \,.\, \vec{\eta} \,\mathrm{d}s \\
&= \int_\Gamma (-\sigma \kappa_{\mathcal{M}} - \mu - \lambda_1 f_1 + \lambda_2 f_2) \, \vec{\nu}_{\mathcal{M}} \,.\, \vec{\eta} \,\mathrm{d}s.
\end{aligned} \tag{4.13}$$

In the second line we used a transport theorem for curves on surfaces, cf. Garcke and Wieland (2006). We applied integration by parts for the second last identity. The last identity follows from (2.46) and $\vec{\eta} \,.\, \vec{\nu}_\phi = 0$.

A time-dependent function $\vec{x} : I \times [0,T] \to \mathcal{M}$ with $\vec{x}_t(\,.\,,t) \in \underline{V}_\Phi$ is called a solution to the gradient flow equation if for all $\vec{\eta} \in \underline{V}_\Phi$

$$(\vec{x}_t, \vec{\eta})_{2,\mathcal{M},nor} = -\left(\delta E(\Gamma)\right)(\vec{\eta}) \tag{4.14}$$

holds.

Let $\eta \in \underline{V}_\Phi$. We conclude from (4.13) and (4.14) on noting that $\vec{\nu}_\mathcal{M} \,.\, \vec{\eta} = \vec{\nu}_\mathcal{M} \,.\, \vec{P}_\mathcal{M}\vec{\eta}$

$$\vec{P}_\mathcal{M}\vec{x}_t = -\left(-\sigma\kappa_\mathcal{M} - \mu - \lambda_1 f_1 + \lambda_2 f_2\right)\vec{\nu}_\mathcal{M}. \tag{4.15}$$

Let $V_n := \vec{x}_t \,.\, \vec{\nu}_\mathcal{M}$ denote the normal velocity. Then $\vec{P}_\mathcal{M}\vec{x}_t = V_n\vec{\nu}_\mathcal{M}$ and consequently the equation above leads to

$$V_n = \sigma\kappa_\mathcal{M} + F, \tag{4.16}$$

where F is given by

$$F(\vec{z}) = \mu + \lambda_1 f_1(\vec{z}) - \lambda_2 f_2(\vec{z}) = \mu + \lambda_1(u_0(\vec{z}) - c_1)^2 - \lambda_2(u_0(\vec{z}) - c_2)^2. \tag{4.17}$$

We rewrite the equation to a scheme for $\vec{x} : I \times [0,T] \to \mathbb{R}^3$ and $\kappa_\mathcal{M}, \kappa_\Phi : I \times [0,T] \to \mathbb{R}$. We assume that $\vec{x}(\rho, 0)$ lies on $\mathcal{M}$. To force the curve to stay on the manifold $\mathcal{M}$, the velocity in direction normal to the surface $\vec{x}_t \,.\, \vec{\nu}_\Phi$ must be zero (i.e. $\vec{x}_t \in \underline{V}_\Phi$). We thus have the following scheme:

Let $\vec{x}(I,0) = \Gamma(0) \subset \mathcal{M}$. For $t \in (0,T]$, find $\vec{x}(\,.\,,t) : I \to \mathbb{R}^3$ and $\kappa_\mathcal{M}(\,.\,,t), \kappa_\Phi(\,.\,,t) : I \to \mathbb{R}$ such that

$$\vec{x}_t \,.\, \vec{\nu}_\mathcal{M} = \sigma\kappa_\mathcal{M} + F, \tag{4.18a}$$

$$\vec{x}_t \,.\, \vec{\nu}_\Phi = 0, \tag{4.18b}$$

$$\vec{x}_{ss} = \kappa_\mathcal{M}\vec{\nu}_\mathcal{M} + \kappa_\Phi\vec{\nu}_\Phi. \tag{4.18c}$$

4.2.3 Multiphase Image Segmentation with Possible Triple Junctions

For multiphase image segmentation, we consider a decomposition of $\mathcal{M}$ in time-dependent regions $\Omega_1(t), \ldots, \Omega_{N_R}(t)$, $t \in [0,T]$, separated by curves $\Gamma_1(t), \ldots, \Gamma_{N_C}(t)$. Each curve is parameterized by a time-dependent function $\vec{x}_i(\,.\,,t) : I_i \to \mathbb{R}^3$, where I_i is a one-dimensional reference manifold for $i = 1, \ldots, N_C$. Similar to the case of one curve, we set $\vec{\nu}_{\Phi,i} = \vec{n}_\Phi \circ \vec{x}_i$ and $\vec{\nu}_{\mathcal{M},i} = (\vec{x}_i)_s \times \vec{\nu}_{\Phi,i}$. The geodesic curvature $\kappa_{\mathcal{M},i}$ and the normal curvature $\kappa_{\Phi,i}$ are given by $\kappa_{\mathcal{M},i} = (\vec{x}_i)_{ss} \,.\, \vec{\nu}_{\mathcal{M},i}$ and $\kappa_{\Phi,i} = (\vec{x}_i)_{ss} \,.\, \vec{\nu}_{\Phi,i}$. All quantities are time-dependent.

We define a piecewise constant image approximation by $u(\,.\,,t) = \sum_{k=1}^{N_R} c_k(t)\chi_{\Omega_k(t)}$, where $\chi_{\Omega_k(t)}$ is the characteristic function on the set $\Omega_k(t)$ and the coefficients $c_k(t)$ are computed by

$$c_k(t) = \frac{\int_{\Omega_k(t)} u_0 \,\mathrm{d}A}{\int_{\Omega_k(t)} 1 \,\mathrm{d}A}, \tag{4.19}$$

i.e. they are set to the mean of u_0 in $\Omega_k(t)$.

Let $\vec{x}_i(\,.\,,0)$, $i = 1, \ldots, N_C$, be parameterizations of given curves $\Gamma_i(0) \subset \mathcal{M}$. We have to solve the following scheme for $t \in (0,T]$: Find $\vec{x}_i(\,.\,,t) : I_i \to \mathbb{R}^3$, $\kappa_{\mathcal{M},i}(\,.\,,t), \kappa_{\Phi,i}(\,.\,,t) : I_i \to \mathbb{R}$ such that

$$(\vec{x}_i)_t \,.\, \vec{\nu}_{\mathcal{M},i} = \sigma\kappa_{\mathcal{M},i} + F_i, \tag{4.20a}$$

$$(\vec{x}_i)_t \,.\, \vec{\nu}_{\Phi,i} = 0, \tag{4.20b}$$

$$(\vec{x}_i)_{ss} = \kappa_{\mathcal{M},i}\,\vec{\nu}_{\mathcal{M},i} + \kappa_{\Phi,i}\,\vec{\nu}_{\Phi,i}, \tag{4.20c}$$

hold for $i = 1, \dots, N_C$. The external force F_i is defined for $\vec{x} \in \mathcal{M}$ by

$$F_i(\vec{x}) = \mu + \lambda_{k^+(i)}(u_0(\vec{x}) - c_{k^+(i)})^2 - \lambda_{k^-(i)}(u_0(\vec{x}) - c_{k^-(i)})^2, \tag{4.21}$$

where $k^+(i), k^-(i) \in \{1, \dots, N_R\}$ are indices of two regions, such that $\vec{\nu}_{\mathcal{M},i}$ points from $\Omega_{k^-(i)}$ to $\Omega_{k^+(i)}$. In the experiments, presented in Section 4.4, we always consider the case $\mu = 0$ and $\lambda_k = \lambda$ for all $k = 1, \dots, N_R$, i.e. all segmentations in our demonstrations can be performed with only two weighting parameters σ and λ.

We also allow open curves, i.e. curves with $\partial\Gamma_i(t) \neq \emptyset$. Since we consider interface curves, i.e. each curve $\Gamma_i(t)$ separates two different regions $\Omega_{k^+(i)}$ and $\Omega_{k^-(i)}$, we exclude free endpoints. Further, we consider only smooth, compact surfaces $\mathcal{M}$ without boundary. Thus, boundary intersections like in the planar case do not occur. Curve endpoints are therefore part of triple junctions denoted with $\vec{\Lambda}_k \in \mathcal{M}$, $k = 1, \dots, N_T$.

For each $k \in \{1, \dots, N_T\}$ let $i_{k,1}, i_{k,2}, i_{k,3} \in \{1, \dots, N_C\}$ denote the indices of curves $\Gamma_{i_{k,l}}$, $l = 1, 2, 3$, $i_{k,1} \neq i_{k,2} \neq i_{k,3} \neq i_{k,1}$ with parameterizations $\vec{x}_{i_{k,l}} : I_{i_{k,l}} = [0, 1] \to \mathcal{M}$, such that

$$\vec{x}_{i_{k,1}}(\rho_{k,1}) = \vec{x}_{i_{k,2}}(\rho_{k,2}) = \vec{x}_{i_{k,3}}(\rho_{k,3}) = \vec{\Lambda}_k,$$

where $\rho_{k,l} \in \{0, 1\}$ corresponds to the start or endpoint of the curve $i_{k,l}$, $l = 1, 2, 3$.

At the triple junctions $\vec{\Lambda}_k$, $k = 1, \dots, N_T$, an attachment condition and Young's law need to hold:

$$\text{the triple junction } \vec{\Lambda}_k \text{ does not pull apart,} \tag{4.22a}$$

$$\sum_{l=1}^{3} (-1)^{\rho_{k,l}} \, \vec{\tau}_{i_{k,l}} = 0, \tag{4.22b}$$

where $\vec{\tau}_{i_{k,l}} := (\vec{x}_{i_{k,l}})_s$ is a tangent vector field at $\Gamma_{i_{k,l}} \subset \mathcal{M}$, $l = 1, 2, 3$. This is analogue to the planar case.

Color images can be handled as in the planar case, recall Section 3.2.9. Again, only the external force need to be adapted compared to scalar images.

4.2.4 Weak Scheme

We introduce a weak scheme for (4.20) with (4.22) in case of triple junctions. We define the following function spaces

$$W := H^1(I_1, \mathbb{R}) \times \ldots \times H^1(I_{N_C}, \mathbb{R}), \tag{4.23a}$$

$$\underline{V} := \big\{ (\vec{\eta}_1, \dots, \vec{\eta}_{N_C}) \in H^1(I_1, \mathbb{R}^3) \times \ldots \times H^1(I_{N_C}, \mathbb{R}^3) : \\ \vec{\eta}_{i_{k,1}}(\rho_{k,1}) = \vec{\eta}_{i_{k,2}}(\rho_{k,2}) = \vec{\eta}_{i_{k,3}}(\rho_{k,3}),\ \forall k = 1, \dots, N_T \big\}, \tag{4.23b}$$

$$\underline{V}_\Phi := \{ (\vec{\eta}_1, \dots, \vec{\eta}_{N_C}) \in \underline{V} \,:\, \vec{\eta}_i \,.\, \vec{\nu}_{\Phi,i} = 0,\ \forall i = 1, \dots, N_C \}. \tag{4.23c}$$

Note, that we re-defined the space $\underline{V}_\Phi$ which now supports the multiphase formulation. Further, we only require functions in $\underline{V}_\Phi$ to be weak differentiable.

For scalar and vector-valued functions $u = (u_1, \dots, u_{N_C})$, $v = (v_1, \dots, v_{N_C}) \in L^2(I_1, \mathbb{R}^{(3)}) \times \ldots \times L^2(I_{N_C}, \mathbb{R}^{(3)})$, we define

$$\int_\Gamma u \,.\, v \, \mathrm{d}s := \sum_{i=1}^{N_C} \int_{I_i} u_i \,.\, v_i \, \|(\vec{x}_i)_\rho\| \, \mathrm{d}\rho.$$

Further, we use the notation

$$\vec{\nu}_{\mathcal{M}} = (\vec{\nu}_{\mathcal{M},1}, \dots, \vec{\nu}_{\mathcal{M},N_C}), \quad \vec{\nu}_{\Phi} = (\vec{\nu}_{\Phi,1}, \dots, \vec{\nu}_{\Phi,N_C})$$

and the short hand notation

$$(\nabla_s u \,.\, \nabla_s v)_{|\Gamma_i} := (u_s \,.\, v_s)_{|\Gamma_i} := (u_i)_s \,.\, (v_i)_s = \frac{(u_i)_\rho \,.\, (v_i)_\rho}{\|(\vec{x}_i)_\rho\|^2}.$$

Now, we consider time-dependent curves. A possible weak scheme is given as follows: Find time-dependent functions $\vec{x}$, $\kappa_{\mathcal{M}}$ and κ_{Φ}, with $\vec{x}(\,.\,,t) \in \underline{V}$, $\vec{x}_t(\,.\,,t) \in \underline{V}$ and $\kappa_{\mathcal{M}}(\,.\,,t), \kappa_{\Phi}(\,.\,,t) \in W$ for $t \in [0,T]$, such that

$$\int_{\Gamma(t)} \vec{x}_t \,.\, \vec{\nu}_{\mathcal{M}}\, \chi \,\mathrm{d}s - \sigma \int_{\Gamma(t)} \kappa_{\mathcal{M}}\, \chi \,\mathrm{d}s = \int_{\Gamma(t)} F\, \chi \,\mathrm{d}s, \qquad \forall \chi \in W, \tag{4.24a}$$

$$\int_{\Gamma(t)} \vec{x}_t \,.\, \vec{\nu}_{\Phi}\, \chi \,\mathrm{d}s = 0, \qquad \forall \chi \in W, \tag{4.24b}$$

$$\int_{\Gamma(t)} (\kappa_{\mathcal{M}}\, \vec{\nu}_{\mathcal{M}} + \kappa_{\Phi}\, \vec{\nu}_{\Phi}) \,.\, \vec{\eta} \,\mathrm{d}s + \int_{\Gamma(t)} \vec{x}_s \,.\, \vec{\eta}_s \,\mathrm{d}s = 0, \qquad \forall \vec{\eta} \in \underline{V}. \tag{4.24c}$$

By (4.24b), we weakly enforce (4.20b). Alternatively, the condition (4.20b) can be incorporated in the function spaces resulting in the following scheme:

Find time-dependent functions $\vec{x}$ and $\kappa_{\mathcal{M}}$, with $\vec{x}(\,.\,,t) \in \underline{V}$, $\vec{x}_t(\,.\,,t) \in \underline{V}_{\Phi}$ and $\kappa_{\mathcal{M}}(\,.\,,t) \in W$ for $t \in [0,T]$, such that

$$\int_{\Gamma(t)} \vec{x}_t \,.\, \vec{\nu}_{\mathcal{M}}\, \chi \,\mathrm{d}s - \sigma \int_{\Gamma(t)} \kappa_{\mathcal{M}}\, \chi \,\mathrm{d}s = \int_{\Gamma(t)} F\, \chi \,\mathrm{d}s, \qquad \forall \chi \in W, \tag{4.25a}$$

$$\int_{\Gamma(t)} \kappa_{\mathcal{M}}\, \vec{\nu}_{\mathcal{M}} \,.\, \vec{\eta} \,\mathrm{d}s + \int_{\Gamma(t)} \vec{x}_s \,.\, \vec{\eta}_s \,\mathrm{d}s = 0, \qquad \forall \vec{\eta} \in \underline{V}_{\Phi}. \tag{4.25b}$$

4.2.5 Image Restoration with Edge Enhancement

Similar to the planar case, we can perform image segmentation and restoration in a two-step approach. Therefore, we consider the Mumford-Shah functional (4.8) for images on surfaces. Let Γ be a union of curves and $u : \mathcal{M} \to [0,1]$ an approximation of the image function $u_0 : \mathcal{M} \to [0,1]$.

For image denoising with smooth functions, we now consider piecewise smooth approximations $u_{|\Omega_k} = u_k$, where $u_k : \Omega_k \to [0,1]$ is a smooth function defined on the region $\Omega_k \subset \mathcal{M}$. We want to perform an image smoothing as a postprocessing step: First, we find a segmentation solving the evolution equations (4.20) with (4.22) derived from the Chan-Vese model. For that, a piecewise constant approximation of u_0 is used. Second, we use the final regions and perform a denoising of the image by solving surface partial differential equations.

To derive the surface PDEs for image denoising, we consider variations of u of the form

$u + \epsilon v$, for $v : \mathcal{M} \to \mathbb{R}$ and $\epsilon > 0$. We compute

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}\epsilon}\bigg|_{\epsilon=0} E^{\mathrm{MS}}(u+\epsilon v, \Gamma) &= \lim_{\epsilon\to 0} \frac{1}{\epsilon}(E^{\mathrm{MS}}(u+\epsilon v,\Gamma) - E^{\mathrm{MS}}(u,\Gamma)) \\
&= \lim_{\epsilon\to 0} \frac{1}{\epsilon}\left(\int_{\mathcal{M}\setminus\Gamma} \left(2\epsilon \nabla_{\mathcal{M}} u \,.\, \nabla_{\mathcal{M}} v + \epsilon^2 \|\nabla_{\mathcal{M}} v\|^2\right) \mathrm{d}A + \right. \\
&\quad \left. + \lambda \int_{\mathcal{M}} \left(2\epsilon(u-u_0)\, v + \epsilon^2 v^2\right) \mathrm{d}A\right) \\
&= 2\int_{\mathcal{M}\setminus\Gamma} \nabla_{\mathcal{M}} u \,.\, \nabla_{\mathcal{M}} v \,\mathrm{d}A + 2\lambda \int_{\mathcal{M}} (u-u_0)\, v \,\mathrm{d}A \\
&= 2\sum_{k=1}^{N_R} \int_{\Omega_k} \left(\nabla_{\mathcal{M}} u_k \,.\, \nabla_{\mathcal{M}} v + \lambda(u_k - u_0)\, v\right) \mathrm{d}A.
\end{aligned}$$

For a stationary solution, we consider $0 = \frac{\mathrm{d}}{\mathrm{d}\epsilon}\big|_{\epsilon=0} E^{\mathrm{MS}}(u+\epsilon v, \Gamma)$. This leads to

$$0 = \int_{\Omega_k} \left(\nabla_{\mathcal{M}} u_k \,.\, \nabla_{\mathcal{M}} v + \lambda(u_k - u_0)\, v\right) \mathrm{d}A$$

for each $k = 1, \ldots, N_R$ and an arbitrary function v which is smooth on Ω_k. Using integration by parts (cf. Dziuk and Elliott (2013) and Section 2), we obtain

$$0 = \int_{\Omega_k} \left(-\Delta_{\mathcal{M}} u_k + \lambda(u_k - u_0)\right) v \,\mathrm{d}A + \int_{\partial\Omega_k} \nabla_{\mathcal{M}} u_k \,.\, \vec{\mu}_k \, v \,\mathrm{d}s \tag{4.26}$$

where $\Delta_{\mathcal{M}}$ is the Laplace-Beltrami operator and $\vec{\mu}_k(\vec{p})$ is a unit outer normal vector on $\partial\Omega_k$ in $T_{\vec{p}}\mathcal{M}$, i.e. it is tangent to the surface for each $\vec{p} \in \partial\Omega_k \subset M$ but normal to $\partial\Omega_k$ in $\vec{p}$. Since $\mathcal{M}$ is a smooth, compact surface without boundary, the boundary $\partial\Omega_k$ of the region Ω_k consists of one or more curves Γ_i, $i \in \{1, \ldots, N_C\}$. Locally, $\vec{\mu}_k$ thus is $\pm\vec{\nu}_{\mathcal{M},i}$. Since v is arbitrary chosen, we have to solve the following surface PDE for $k = 1, \ldots, N_R$:

$$-\frac{1}{\lambda}\Delta_{\mathcal{M}} u_k + u_k = u_0, \quad \text{in } \Omega_k, \tag{4.27a}$$

with Neumann boundary conditions

$$\nabla_{\mathcal{M}} u_k \,.\, \vec{\mu}_k = 0, \quad \text{on } \partial\Omega_k. \tag{4.27b}$$

This is the analogue to the planar diffusion scheme (3.43), where the Laplace operator is replaced by the Laplace-Beltrami operator and the gradient in the Neumann boundary condition is replaced by the surface gradient.

4.2.6 Related Works

The works on evolution of curves on surfaces and on active contours for images on surfaces differ in the way how the surface and the curves are described mathematically.

A geometric scale space for images on parametric surfaces is introduced by Kimmel (1997) using level sets. The image is handled implicitly by considering its iso-gray levels.

Spira and Kimmel (2007) consider flows of curves on parametric surfaces and perform edge detection using the geodesic active contours method. They restrict on surfaces which have a global parameterization. They solve evolution equations with the level set method by

considering the pre-image of the curve resulting in a planar curve in the parameterization plane. However, their approach has intrinsic disadvantages since they use the pre-image of the curve. Krüger et al. (2008) study drawbacks of this approach concerning the scaling behavior. Further, the method does not allow to incorporate a balloon term (recall (3.14)), see Krüger et al. (2008).

In Cheng et al. (2002), the surface is represented by the zero level set of a three-dimensional function. The zero level set of a second three-dimensional function, which is time-dependent, is used additionally. The curve is represented by the intersection of the two zero level sets. This approach is used by Krüger et al. (2008) and Zhou and Li (2013) for image segmentation with geodesic active contours. A drawback of the method is that a one-dimensional curve evolution problem is extended to a three-dimensional problem. To reduce the effort, Krüger et al. (2008) propose a new narrow band technique.

Tian et al. (2009) perform image segmentation using the Chan-Vese model for images on surfaces with level set methods in combination with a so-called closest point method. They compute one iteration step of the Chan-Vese model in a small 3D neighborhood of the surface followed by an interpolation step.

Mikula and Ševčovič (2006) study flows of the form $V_n = \kappa_{\mathcal{M}} + F$, where F is an external forcing term. The authors restrict on graph surfaces and solve a system of partial differential equations for the parameterization $\vec{x}$, the tangent angle, the geodesic curvature $\kappa_{\mathcal{M}}$ and for $\|\vec{x}_\rho\|$.

In this thesis, we make use of the concept developed by Barrett et al. (2010a), where the authors consider flows of curves on surfaces like geodesic curvature flow, geodesic surface diffusion and geodesic Willmore flow of curves. Here, we apply the methods of Barrett et al. (2010a) to image segmentation applications resulting in a mean curvature related flow: The normal velocity is the sum of the geodesic curvature $\kappa_{\mathcal{M}}$ weighted with a parameter σ and an external forcing term. The forcing term is designed for image segmentation based on an extension of the Chan-Vese method to images on surfaces. As in Barrett et al. (2010a) a parametric approach is used to describe the curve, recall scheme (4.20). The surface is given implicitly, as zero level set of a smooth function Φ. However, for practical image segmentation, the function Φ need not be explicitly known. Our method requires only a normal vector to the surface at each point $\vec{p} \in \mathcal{M}$.

Furthermore, for image denoising we solve a diffusion equation on the surface $\mathcal{M}$. For related works on surface partial differential equations, we refer to Dziuk and Elliott (2013) and the references therein. Smooth surfaces are approximated by triangulated surfaces to solve surface PDEs numerically. For a practical application see Čunderlík et al. (2013), where data of the Earth's surface captured on board of satellites are filtered by nonlinear diffusion. An implicit representation of the surface is used in Bertalmio et al. (2001), where the surface is embedded as zero level set of a higher dimensional function and the partial differential equations are solved on a fixed Cartesian grid using a special embedding function.

Lai and Chan (2011) propose methods to solve total variation problems on surfaces. In detail, they generalize the method of Rudin et al. (1992) for denoising images on surfaces and the method of Chan et al. (2006) (a convexified Chan-Vese model) for segmentation of images on surfaces. They pursue a direct, called *intrinsic*, approach: they perform the resulting calculations directly on the given surface by using differential geometry techniques and finite elements for images given on triangulated surfaces. Lai and Chan (2011) also provide an overview and a comparison of several approaches for variational problems on surfaces (level set methods, parametric surfaces and direct/intrinsic methods). Total variation based image

restoration and segmentation are also considered by Wu et al. (2012), where the authors propose an extension of the augmented Lagrangian method (see e.g. Wu and Tai (2010)) for scalar and vectorial total variation problems to images on surfaces.

4.3 Numerical Approximation

4.3.1 Finite Element Approximation

We introduce a finite element approximation for the scheme (4.20) with (4.22) in the case of triple junctions. The finite element scheme is based on the weak formulation (4.25). The scheme (4.20) can be interpreted as weighted geodesic curvature flow with an external forcing term. Therefore, we make use of the finite element scheme for geodesic curvature flow developed by Barrett et al. (2010a) for closed curves and generalize the approach for possible open curves with triple junctions and for image segmentation problems.

Spatial and Time Discretization

Similar to the planar case, we introduce a spatial and time discretization, introduce discrete function spaces and discrete inner products. We use a similar notation as in the planar case, where the vector-valued quantities lie now in $\mathbb{R}^3$ instead of $\mathbb{R}^2$. A restriction to $\mathcal{M}$ is achieved by a discrete version of the space $\underline{V}_\Phi$.

For $i = 1, \dots, N_C$, let $0 = q_0^i < q_1^i < \dots < q_{N_i}^i = 1$ be a decomposition of the interval $I_i = [0,1]$. If Γ_i is a closed curve, we make use of the periodicity $N_i = 0$, $N_i + 1 = 1$, $-1 = N_i - 1$, etc.

Let N_T be the number of triple junctions. For $k = 1, \dots, N_T$, let $\vec{\Lambda}_k \in \mathcal{M}$ denote a triple point. Let $i_{k,l}$, $l = 1,2,3$, be the indices of curves and $\rho_{k,l} \in \{0,1\}$ such that $\vec{x}_{i_{k,1}}(\rho_{k,1}) = \vec{x}_{i_{k,2}}(\rho_{k,2}) = \vec{x}_{i_{k,3}}(\rho_{k,3}) = \vec{\Lambda}_k$.

We introduce the following discrete function spaces

$$
\begin{aligned}
W^h := \Big\{ &(\eta_1, \dots, \eta_{N_C}) \in C(I_1, \mathbb{R}) \times \dots \times C(I_{N_C}, \mathbb{R}) \,:\, \eta_i|_{[q_{j-1}^i, q_j^i]} \text{ is linear,} \\
&\forall i = 1, \dots, N_C,\, j = 1, \dots, N_i \Big\}, \qquad (4.28a) \\
\underline{V}^h := \Big\{ &(\vec{\eta}_1, \dots, \vec{\eta}_{N_C}) \in C(I_1, \mathbb{R}^3) \times \dots \times C(I_{N_C}, \mathbb{R}^3) \,:\, \vec{\eta}_{i_{k,1}}(\rho_{k,1}) = \\
&= \vec{\eta}_{i_{k,2}}(\rho_{k,2}) = \vec{\eta}_{i_{k,3}}(\rho_{k,3}),\, \forall k = 1, \dots, N_T,\, \vec{\eta}_i|_{[q_{j-1}^i, q_j^i]} \text{ is linear,} \\
&\forall i = 1, \dots, N_C,\, j = 1, \dots, N_i \Big\}. \qquad (4.28b)
\end{aligned}
$$

A basis of W^h is given by functions $\chi_{i,j} := ((\chi_{i,j})_1, \dots, (\chi_{i,j})_{N_C}) \in W^h$, where $(\chi_{i,j})_k(q_l^k) := \delta_{ik}\delta_{jl}$ for $i,k = 1, \dots, N_C$, $j = j_0^i, \dots, N_i$, $l = j_0^k, \dots, N_k$, where $j_0^i = 1$ if Γ_i is closed and $j_0^i = 0$ else. We call $\{\chi_{i,j}\}_{i=1,\dots,N_C, j=j_0^i,\dots,N_i}$ standard basis of W^h.

Further, let $0 = t_0 < t_1 < \dots < t_M = T$ be a partitioning of the time interval $[0,T]$ into possibly variable time steps $\tau_m := t_{m+1} - t_m$, $m = 0, \dots, M-1$.

Let $\vec{X}^m = (\vec{X}_1^m, \dots, \vec{X}_{N_C}^m) \in \underline{V}^h$ be an approximation of $\vec{x}(\,.\,, t_m) = (\vec{x}_1(\,.\,, t_m), \dots, \vec{x}_{N_C}(\,.\,, t_m))$. Let $\Gamma^m = (\Gamma_1^m, \dots, \Gamma_{N_C}^m)$ denote the image of $\vec{X}^m$.

In case of triple junctions let $j_{k,l} \in \{0, N_{i_{k,l}}\}$ denote the index of the corresponding curve endpoint such that $q^{i_{k,l}}_{j_{k,l}} = \rho_{k,l}$, $k = 1, \dots, N_T$, $l = 1, 2, 3$.

For scalar or vector functions $u = (u_1, \dots, u_{N_C}), v = (v_1, \dots, v_{N_C}) \in L^2(I_1, \mathbb{R}^{(3)}) \times \dots \times L^2(I_{N_C}, \mathbb{R}^{(3)})$, the L^2-inner product $\langle \cdot\,, \cdot \rangle_m$ over the current polygonal curve network Γ^m is given by

$$\langle u, v \rangle_m := \int_{\Gamma^m} u \,.\, v \,\mathrm{d}s := \sum_{i=1}^{N_C} \int_{I_i} u_i \,.\, v_i \, \|(\vec{X}_i^m)_\rho\| \,\mathrm{d}\rho. \tag{4.29}$$

Further, we make use of a mass lumped inner product as in the planar case: The mass lumped inner product $\langle \cdot\,, \cdot \rangle^h_m$ for piecewise continuous functions $u = (u_1, \dots, u_{N_C})$ and $v = (v_1, \dots, v_{N_C})$ is defined by

$$\langle u, v \rangle^h_m := \frac{1}{2} \sum_{i=1}^{N_C} \sum_{j=1}^{N_i} h^m_{i,j-\frac{1}{2}} \left[(u_i \,.\, v_i) \left([q^i_j]^- \right) + (u_i \,.\, v_i) \left([q^i_{j-1}]^+ \right) \right], \tag{4.30}$$

where $u_i([q^i_j]^\pm) := \lim_{\epsilon \to 0, \epsilon > 0} u_i(q^i_j \pm \epsilon)$ and $h^m_{i,j-\frac{1}{2}} := \|\vec{X}^m_i(q^i_j) - \vec{X}^m_i(q^i_{j-1})\| > 0$ is the distance between two neighbor nodes. Let $h^m := \max_{i=1,\dots,N_C, j=1,\dots,N_i} h^m_{i,j-\frac{1}{2}}$ denote the maximum distance between two neighbor nodes of the polygonal curves.

Discrete Vector Fields

Let $\vec{X}^m \in \underline{V}^h$ be a given parameterization of the polygonal curve network Γ^m satisfying the following assumption:

($\mathcal{A}$) The distance between two neighbor nodes of Γ^m is positive, i.e. $h^m_{i,j-\frac{1}{2}} > 0$ for $i = 1, \dots, N_C$, $j = 1, \dots, N_i$ and $\vec{X}^m_i(q^i_{j+1}) \neq \vec{X}^m_i(q^i_{j-1})$ for $i = 1, \dots, N_C$ and $j = 1, \dots, N_i$ if $\partial\Gamma^m_i = \emptyset$ and $j = 1, \dots, N_i - 1$ if $\partial\Gamma^m_i \neq \emptyset$.

We define $\vec{\omega}^m_\Phi = (\vec{\omega}^m_{\Phi,1}, \dots, \vec{\omega}^m_{\Phi,N_C})$, $\vec{\omega}^m_d = (\vec{\omega}^m_{d,1}, \dots, \vec{\omega}^m_{d,N_C})$ and $\vec{\omega}^m_{\mathcal{M}} = (\vec{\omega}^m_{\mathcal{M},1}, \dots, \vec{\omega}^m_{\mathcal{M},N_C})$ by

$$\vec{\omega}^m_{\Phi,i}(q^i_j) = \vec{n}_\Phi(\vec{X}^m_i(q^i_j)) = \frac{\nabla\Phi(\vec{X}^m_i(q^i_j))}{\|\nabla\Phi(\vec{X}^m_i(q^i_j))\|}, \tag{4.31a}$$

$$\vec{\omega}^m_{d,i}(q^i_j) = \begin{cases} \frac{\vec{X}^m_i(q^i_{j+1}) - \vec{X}^m_i(q^i_{j-1})}{\|\vec{X}^m_i(q^i_{j+1}) - \vec{X}^m_i(q^i_{j-1})\|}, & \text{if } \partial\Gamma^m_i = \emptyset, \text{ or if } \partial\Gamma^m_i \neq \emptyset \text{ and } j \neq 0, N_i, \\ \frac{\vec{X}^m_i(q^i_1) - \vec{X}^m_i(q^i_0)}{\|\vec{X}^m_i(q^i_1) - \vec{X}^m_i(q^i_0)\|}, & \text{if } \partial\Gamma^m_i \neq \emptyset, \text{ and } j = 0, \\ \frac{\vec{X}^m_i(q^i_{N_i}) - \vec{X}^m_i(q^i_{N_i-1})}{\|\vec{X}^m_i(q^i_{N_i}) - \vec{X}^m_i(q^i_{N_i-1})\|}, & \text{if } \partial\Gamma^m_i \neq \emptyset, \text{ and } j = N_i. \end{cases} \tag{4.31b}$$

$$\vec{\omega}^m_{\mathcal{M},i}(q^i_j) = \vec{\omega}^m_{d,i}(q^i_j) \times \vec{\omega}^m_{\Phi,i}(q^i_j), \tag{4.31c}$$

for $i = 1, \dots, N_C$ and $j = j^i_0, \dots, N_i$. For closed curves, we make use of the periodicity $N_i = 0$, $N_i + 1 = 1$ and $-1 = N_i - 1$.

The assumption ($\mathcal{A}$) is needed, such that $\vec{\omega}^m_d$ is well-defined, see also Barrett et al. (2010a).

We define a discrete analogue to the space $\underline{V}_\Phi$ by

$$\underline{V}^h_\Phi = \left\{ \vec{\eta} \in \underline{V}^h \,:\, \vec{\eta}_i \,.\, \vec{\omega}^m_{\Phi,i} = 0, \quad i = 1, \dots, N_C \right\}. \tag{4.32}$$

Discrete scheme

We propose the following discrete scheme: Let $\vec{X}^0 \in \underline{V}^h$ be a given parameterization of a polygonal curve network Γ^0. We assume that the initial nodes $\vec{X}_i^0(q_j^i)$ lie on the surface $\mathcal{M}$. Further, we assume that assumption $(\mathcal{A})$ holds for $\vec{X}^m$, $m = 0, \dots, M-1$.

For $m = 0, \dots, M-1$, find $\delta\vec{X}^{m+1} \in \underline{V}_\Phi^h$ and $\kappa_{\mathcal{M}}^{m+1} \in W^h$ such that

$$\langle \frac{\delta\vec{X}^{m+1}}{\tau_m}, \chi\,\vec{\omega}_{\mathcal{M}}^m \rangle_m^h - \sigma\langle \kappa_{\mathcal{M}}^{m+1}, \chi\rangle_m^h = \langle F^m, \chi\rangle_m^h, \qquad \forall \chi \in W^h, \tag{4.33a}$$

$$\langle \kappa_{\mathcal{M}}^{m+1}\,\vec{\omega}_{\mathcal{M}}^m, \vec{\eta}\rangle_m^h + \langle \nabla_s \delta\vec{X}^{m+1}, \nabla_s\vec{\eta}\rangle_m = -\langle \nabla_s\vec{X}^m, \nabla_s\vec{\eta}\rangle_m, \qquad \forall \vec{\eta} \in \underline{V}_\Phi^h, \tag{4.33b}$$

where $F^m = (F_1^m, \dots, F_{N_C}^m) \in W^h$, with F_i^m, $i = 1, \dots, N_C$, is the piecewise linear function uniquely given by

$$F_i^m(q_j^i) = \lambda\left[\left(u_0(\vec{X}_i^m(q_j^i)) - c_{k^+(i)}^m\right)^2 - \left(u_0(\vec{X}_i^m(q_j^i)) - c_{k^-(i)}^m\right)^2\right], \tag{4.34}$$

where $c_{k^\pm(i)}^m$ are approximations of the coefficients $c_{k^\pm(i)}(t_m)$, cf. (4.19). We will later state how the coefficients c_k^m, $k = 1, \dots, N_R$, are computed in practice.

Having found $\delta\vec{X}^{m+1} \in \underline{V}_\Phi^h$, we set $\vec{X}^{m+1} := \delta\vec{X}^{m+1} + \vec{X}^m \in \underline{V}^h$.

The finite element scheme (4.33) is the discrete analogue of (4.25).

Before we proceed to prove existence and uniqueness of a solution of the scheme (4.33), we state some very mild assumptions, similar as in the two-dimensional case.

$(\mathcal{A}_1)$ Let $i \in \{1, \dots, N_C\}$. If $\partial\Gamma_i^m = \emptyset$, we assume that

$$\dim \operatorname{span}\{\vec{\omega}_{\mathcal{M},i}^m(q_j^i), \vec{\omega}_{\Phi,i}^m(q_j^i)\}_{j=1}^{N_i} = 3.$$

$(\mathcal{A}_2)$ For each $k \in \{1, \dots, N_T\}$, we assume that

$$\dim \operatorname{span}\{\{\vec{\omega}_{\mathcal{M},i_{k,l}}^m(q_j^{i_{k,l}}), \vec{\omega}_{\Phi,i_{k,l}}^m(q_j^{i_{k,l}})\}_{j=1}^{N_{i_{k,l}}-1}\}_{l=1}^3 = 3.$$

Theorem 4.2. *Let the assumptions $(\mathcal{A})$, $(\mathcal{A}_1)$ and $(\mathcal{A}_2)$ hold. Then there exists a unique solution $(\delta\vec{X}^{m+1}, \kappa_{\mathcal{M}}^{m+1}) \in \underline{V}_\Phi^h \times W^h$ to the system (4.33).*

Proof. The system (4.33) is linear. Therefore, existence of a solution follows from its uniqueness. To prove uniqueness, consider the following system: Find $\{\vec{X}, \kappa_{\mathcal{M}}\} \in \underline{V}_\Phi^h \times W^h$ such that

$$\langle \vec{X}, \chi\,\vec{\omega}_{\mathcal{M}}^m\rangle_m^h - \sigma\tau_m\langle \kappa_{\mathcal{M}}, \chi\rangle_m^h = 0, \qquad \forall \chi \in W^h, \tag{4.35a}$$

$$\langle \kappa_{\mathcal{M}}\,\vec{\omega}_{\mathcal{M}}^m, \vec{\eta}\rangle_m^h + \langle \nabla_s\vec{X}, \nabla_s\vec{\eta}\rangle_m = 0, \qquad \forall \vec{\eta} \in \underline{V}_\Phi^h. \tag{4.35b}$$

We obtain choosing $\chi = \kappa_{\mathcal{M}} \in W^h$ in (4.35a) and $\vec{\eta} = \vec{X} \in \underline{V}_\Phi^h$ in (4.35b)

$$\sigma\tau_m\langle \kappa_{\mathcal{M}}, \kappa_{\mathcal{M}}\rangle_m^h + \langle \nabla_s\vec{X}, \nabla_s\vec{X}\rangle_m = 0.$$

From this equation, we conclude $\kappa_{\mathcal{M}} \equiv 0$ and $\vec{X} \equiv \vec{X}^c$ for a constant $\vec{X}^c = (\vec{X}_1^c, \dots, \vec{X}_{N_C}^c) \in (\mathbb{R}^3)^{N_C}$ with $\vec{X}_{i_{k,1}}^c = \vec{X}_{i_{k,2}}^c = \vec{X}_{i_{k,3}}^c$ for all $k \in \{1, \dots, N_T\}$. Further, $\vec{X}_i^c \in \mathbb{R}^3$ satisfies

$$\vec{X}_i^c \,.\, \vec{\omega}_{\Phi,i}^m(q_j^i) = 0 \tag{4.36}$$

for all $i = 1, \dots, N_C$ and $j = j_0^i, \dots, N_i$, since $\vec{X} \in \underline{V}_\Phi^h$. Inserting $\kappa \equiv 0$ and $\vec{X} \equiv \vec{X}^c$, (4.35a) reduces to

$$\langle \vec{X}^c, \chi\, \vec{\omega}_{\mathcal{M}}^m \rangle_m^h = 0, \quad \forall \chi \in W^h. \tag{4.37}$$

We now choose $\chi = \chi_{i,j} \in W^h$, $i \in \{1, \dots, N_C\}$, $j \in \{j_0^i, \dots, N_i\}$ in (4.37). This yields

$$\vec{X}_i^c \,.\, \vec{\omega}_{\mathcal{M},i}^m(q_j^i) = 0. \tag{4.38}$$

Similar as in the planar case (cf. Theorem 3.5), we can conclude $\vec{X}_i^c \equiv 0$ using (4.36), (4.38) and the assumptions $(\mathcal{A}_1)$ and $(\mathcal{A}_2)$. □

4.3.2 Solution of the Discrete System

Let $N = \sum_{i=1}^{N_C} N_i^*$, with $N_i^* = N_i$ for closed curves and $N_i^* = N_i + 1$ for open curves. We make use of a small abuse of notation and consider functions in W^h as elements in $\mathbb{R}^N$ and functions in $\underline{V}^h$ as elements in

$$\mathbb{X} = \left\{ (\vec{z}_1, \dots, \vec{z}_{N_C}) \in (\mathbb{R}^3)^N \,:\, [\vec{z}_{i_{k,1}}]_{j_{k,1}} = [\vec{z}_{i_{k,2}}]_{j_{k,2}} = [\vec{z}_{i_{k,3}}]_{j_{k,3}},\, k = 1, \dots N_T \right\},$$

where $\vec{z}_i \in (\mathbb{R}^3)^{N_i^*}$ and $[\vec{z}_i]_j \in \mathbb{R}^3$ is the j-th component of the vector $\vec{z}_i$. Functions in $\underline{V}_\Phi^h$ are considered as elements in

$$\mathbb{X}_\Phi = \left\{ (\vec{z}_1, \dots, \vec{z}_{N_C}) \in \mathbb{X} \,:\, [\vec{z}_i]_j \,.\, \vec{\omega}_{\Phi,i}^m(q_j^i) = 0,\, i = 1, \dots, N_C,\, j = j_0^i, \dots, N_i \right\},$$

with $j_0^i = 0$ for open curves and $j_0^i = 1$ for closed curves. Let $\vec{P}_\Phi : (\mathbb{R}^3)^N \to \mathbb{X}_\Phi$ denotes the orthogonal projection onto the space $\mathbb{X}_\Phi$.

In order to state a matrix formulation for the discrete system (4.33), we introduce the following matrices

$$M := \begin{pmatrix} M^1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & M^{N_C} \end{pmatrix}, \quad \vec{N}_{\mathcal{M}} := \begin{pmatrix} \vec{N}_{\mathcal{M}}^1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \vec{N}_{\mathcal{M}}^{N_C} \end{pmatrix}, \quad \vec{A} := \begin{pmatrix} \vec{A}^1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \vec{A}^{N_C} \end{pmatrix},$$

where $M^i \in \mathbb{R}^{N_i^* \times N_i^*}$, $\vec{N}_{\mathcal{M}}^i \in (\mathbb{R}^3)^{N_i^* \times N_i^*}$, $\vec{A}^i \in (\mathbb{R}^{3\times 3})^{N_i^* \times N_i^*}$, $i = 1, \dots, N_C$, are defined by

$$M_{jl}^i := \langle \chi_{i,j}, \chi_{i,l} \rangle_m^h, \quad (\vec{N}_{\mathcal{M}}^i)_{jl} := \langle \chi_{i,j}, \chi_{i,l}\, \vec{\omega}_{\mathcal{M}}^m \rangle_m^h, \quad \vec{A}_{jl}^i := \langle \nabla_s \chi_{i,j}, \nabla_s \chi_{i,l} \rangle_m \, \vec{\mathrm{Id}}_3,$$

where $\vec{\mathrm{Id}}_3$ denotes the 3×3 identity matrix.

We define similar to the planar case $b^m = (b_1^m, \dots, b_{N_C}^m) \in \mathbb{R}^N$ by

$$b_i^m = (b_{i,j_0^i}^m, \dots, b_{i,N_i}^m), \quad \text{with } \; b_{i,j}^m := \langle F_i^m, \chi_{i,j} \rangle_m^h, \; j = j_0^i, \dots, N_i, \tag{4.39}$$

for $i = 1, \dots, N_C$.

The system (4.33) can be rewritten into the following matrix-vector formulation: Find $\kappa_{\mathcal{M}}^{m+1} \in \mathbb{R}^N$ and $\delta\vec{X}^{m+1} \in \mathbb{X}_\Phi$, such that

$$\begin{pmatrix} -\sigma\tau_m M & \vec{N}_{\mathcal{M}}^T \vec{P}_\Phi \\ \vec{P}_\Phi \vec{N}_{\mathcal{M}} & \vec{P}_\Phi \vec{A} \vec{P}_\Phi \end{pmatrix} \begin{pmatrix} \kappa_{\mathcal{M}}^{m+1} \\ \delta\vec{X}^{m+1} \end{pmatrix} = \begin{pmatrix} \tau_m b^m \\ -\vec{P}_\Phi \vec{A} \vec{X}^m \end{pmatrix}, \tag{4.40}$$

holds, on assuming that $\vec{X}^0 \in \mathbb{X}$.

Since M is non-singular, we can apply a Schur complement approach and obtain

$$\kappa_{\mathcal{M}}^{m+1} = \frac{1}{\sigma\tau_m} M^{-1} \left(\vec{N}_{\mathcal{M}}^T \vec{P}_\Phi \, \delta\vec{X}^{m+1} - \tau_m b^m \right), \tag{4.41a}$$

$$\left(\vec{P}_\Phi \vec{A} \vec{P}_\Phi + \frac{1}{\sigma\tau_m} \vec{P}_\Phi \vec{N}_{\mathcal{M}} M^{-1} \vec{N}_{\mathcal{M}}^T \vec{P}_\Phi \right) \delta\vec{X}^{m+1} = \frac{1}{\sigma} \vec{P}_\Phi \vec{N}_{\mathcal{M}} M^{-1} b^m - \vec{P}_\Phi \vec{A} \vec{X}^m, \tag{4.41b}$$

Since $\vec{P}_\Phi$ is a projection to a subspace of $(\mathbb{R}^3)^N$, the system matrix of the linear equation (4.41b) is singular as a mapping of $(\mathbb{R}^3)^N \to (\mathbb{R}^3)^N$. However, considered as a mapping of $\mathbb{X}_\Phi \to \mathbb{X}_\Phi$ it is non-singular if the assumptions $(\mathcal{A}_1)$ and $(\mathcal{A}_2)$ hold.

Since the system matrix is sparse, (4.41b) can be efficiently solved with an iterative solver (with possible preconditioning) or with a direct solver for sparse matrices.

4.3.3 Semidiscrete Scheme

Similar to the planar case, cf. Section 3.3.4, we consider a scheme which is discrete in space and continuous in time. We consider time-dependent polygonal curves $\Gamma_i(t)$, $i = 1, \dots, N_C$, $t \in [0, T]$, and show an equidistribution property concerning the distribution of mesh points along the curves on surfaces.

We make use of a similar notation as in the fully discrete case by just omitting the superscripts m and $m+1$. In detail, let $\vec{X} = (\vec{X}_1, \dots, \vec{X}_{N_C})$ and $\kappa = (\kappa_1, \dots, \kappa_{N_C})$ with $\vec{X}_i : I_i \times [0, T] \to \mathbb{R}^3$, and $\kappa_i : I_i \times [0, T] \to \mathbb{R}$, $i = 1, \dots, N_C$, such that $\vec{X}(\,.\,, t)$ and $\kappa(\,.\,, t)$ are piecewise linear on $[q_{j-1}^i, q_j^i]$, $j = 1, \dots, N_i$, for each $t \in [0, T]$. Further, we set $\vec{X}_{i,j} = \vec{X}_i(q_j^i)$ and $h_{i,j-\frac{1}{2}} = \|\vec{X}_{i,j} - \vec{X}_{i,j-1}\|$ for $i = 1, \dots, N_C$, $j = 1, \dots, N_i$. We will make use of inner products $\langle\,.\,,.\,\rangle$ and $\langle\,.\,,.\,\rangle^h$ which are defined as in (4.29) and (4.30) by replacing $\vec{X}^m$ by $\vec{X}(\,.\,, t)$ and Γ_i^m by the current polygonal curve $\Gamma_i(t)$, $i = 1, \dots, N_C$, $t \in [0, T]$. We make use of vector fields $\vec{\omega}_\Phi$, $\vec{\omega}_d$ and $\vec{\omega}_{\mathcal{M}}$ which are defined similar as in (4.31) by omitting the superscript m. The space $\underline{V}_\Phi^h$ is defined as in (4.32) by using $\vec{\omega}_\Phi$ instead of $\vec{\omega}_\Phi^m$.

Lemma 4.3. *(See also Barrett et al. (2010a)) The semidiscrete scheme*

$$\langle \vec{X}_t, \chi \, \vec{\omega}_{\mathcal{M}} \rangle^h - \sigma \langle \kappa_{\mathcal{M}}, \chi \rangle^h = \langle F, \chi \rangle^h, \qquad \forall \chi \in W^h, \tag{4.42a}$$

$$\langle \kappa_{\mathcal{M}} \, \vec{\omega}_{\mathcal{M}}, \vec{\eta} \rangle^h + \langle \nabla_s \vec{X}, \nabla_s \vec{\eta} \rangle = 0, \qquad \forall \vec{\eta} \in \underline{V}_\Phi^h, \tag{4.42b}$$

for $\kappa \in W^h$ and $\vec{X} \in \underline{V}_\Phi^h$, provides an equidistribution of the mesh points along the curves $\Gamma_i(t)$, $i = 1, \dots, N_C$, $t \in [0, T]$.

Proof. Testing (4.42b) with $\vec{\eta} = \chi_{i,j} \, \vec{\omega}_{d,i}(q_j^i)$, $i = 1, \dots, N_C$, $j = 1, \dots, N_i$ if $\partial\Gamma_i(t) = \emptyset$ and $j = 1, \dots, N_i - 1$ else, leads to

$$0 = \left(\frac{\vec{X}_{i,j+1} - \vec{X}_{i,j}}{h_{i,j+\frac{1}{2}}} - \frac{\vec{X}_{i,j} - \vec{X}_{i,j-1}}{h_{i,j-\frac{1}{2}}} \right) \cdot \left(\vec{X}_{i,j+1} - \vec{X}_{i,j-1} \right), \tag{4.43}$$

where we use that $\vec{\omega}_{\mathcal{M}}$ and $\vec{\omega}_d$ are perpendicular. As in Lemma 3.6, we conclude

$$\|\vec{X}_{i,j+1} - \vec{X}_{i,j}\| = \|\vec{X}_{i,j} - \vec{X}_{i,j-1}\| \quad \text{or} \quad (\vec{X}_{i,j+1} - \vec{X}_{i,j}) \parallel (\vec{X}_{i,j} - \vec{X}_{i,j-1}). \tag{4.44}$$

Consequently, three neighbor nodes are equally distributed or lie on one straight line. □

Remark 4.4. *Note, that we could show that the Euclidean distance between mesh points is equal (or three neighbor nodes lie on one straight line). We did not show an equidistribution using the geodesic distance of the nodes on the surface. However, the distance between neighboring nodes on the polygonal curves is usually very small since the polygonal curves approximate smooth curves. Therefore, if the mesh points are equally distributed with respect to the Euclidean distance, we will also obtain a good mesh quality concerning the distribution of nodes with respect to the geodesic distance.*

As in the planar case, see Section 3.3.4, we cannot prove equidistribution for the fully discrete scheme. However in practical experiments, we observe a good mesh quality during the evolution of the curves on the surface.

4.3.4 Topology Changes

In the introduction of this chapter, we discussed the problem of possible false detections of topology changes, cf. Figure 4.1. If we extend the algorithm to detect topology changes from planar curves, cf. Section 3.3.5, to curves on surfaces, we need to choose the grid size a of the auxiliary background grid carefully.

For $\vec{p} \in \mathcal{M}$ let $T_{\vec{p}}\mathcal{M}$ denote the tangent space and $N_{\vec{p}}\mathcal{M} = (T_{\vec{p}}\mathcal{M})^{\perp}$ the normal space. Let $N\mathcal{M} = \left\{(\vec{p}, \vec{n}) \,:\, \vec{p} \in \mathcal{M},\, \vec{n} \in N_{\vec{p}}\mathcal{M}\right\}$ denote the normal bundle.

For a smooth embedded hypersurface, we consider the map

$$E : N\mathcal{M} \to \mathbb{R}^3, \quad (\vec{p}, \vec{n}) \mapsto \vec{p} + \vec{n}.$$

Theorem 4.5 (Tubular neighborhood theorem)**.** *Every embedded hypersurface $\mathcal{M}$ of $\mathbb{R}^3$ has a tubular neighborhood, i.e. a neighborhood $U \subset \mathbb{R}^3$ that is the diffeomorphic image under $E : N\mathcal{M} \to \mathbb{R}^3$ of an open subset $V \subset N\mathcal{M}$ of the form*

$$V = \{(\vec{p}, \vec{n}) \in N\mathcal{M} \,:\, \|\vec{n}\| < \delta(\vec{p})\}, \tag{4.45}$$

for some positive continuous function $\delta : \mathcal{M} \to \mathbb{R}$.

Proof. See Lee (2002), Chapter 6, Embedding and Approximation Theorems, or Lang (2002), Chapter 4, Vector Fields and Differential Equations. □

We assume that $\mathcal{M}$ is a compact, smooth, embedded hypersurface. As a consequence, set $\delta_0 = \min\{\delta(\vec{p}) \,:\, \vec{p} \in \mathcal{M}\} > 0$. For each $\vec{p} \in \mathcal{M}$ the intersection $B_{\delta_0}(\vec{p}) \cap \mathcal{M}$ is simply connected, which is a consequence of the fact that $E|_V : V \to U$ is a diffeomorphism. An illustration of a surface with a possible tubular neighborhood is presented in Figure 4.2.

Topology changes are now detected as follows:

- Construct an underlying 3D grid with grid size a with $a\sqrt{3} < \delta_0$. Note that the intersection of a grid element (=cube of grid length a) with $\mathcal{M}$ is simply connected.
- Mark the grid elements with the indices of the curves and the mesh points, similarly as in the 2D case.
- If two non-neighbor points $\vec{X}^m_{i,j}$ and $\vec{X}^m_{i_1,j_1}$ lie in the same cube, a topology change is detected.

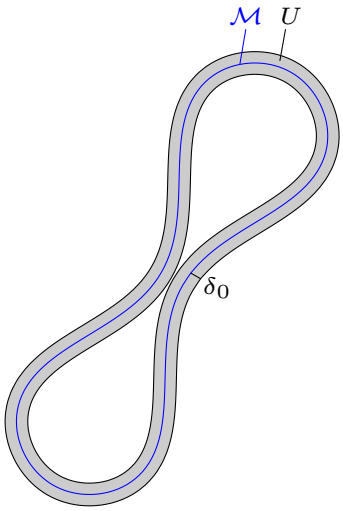


Figure 4.2: Cross-section of a surface $\mathcal{M}$ with a possible tubular neighborhood U.

The key idea is to choose the grid size a of the auxiliary 3D background grid small enough with respect to δ_0, such that points from two different parts of the surface (with nearly opposite normal vector $\vec{\nu}_\Phi$) cannot lie in one array of the grid.

Splitting, merging and the creation of triple junctions are distinguished as for planar curves (cf. Section 3.3.5) by considering the indices of the curves and the indices of the regions separated by the curves.

4.3.5 Additional Computational Aspects

Triangulated surfaces

In practical applications, a smooth function $\Phi : \mathbb{R}^3 \to \mathbb{R}$, such that $\mathcal{M}$ is the zero level set of Φ, is usually not provided. Moreover, a surface $\mathcal{M}$ is typically given as a triangulated surface instead of a smooth surface.

Therefore, we assume that $\mathcal{M}$ is a union of triangles of a triangulation $\mathcal{T}^h$, i.e. $\mathcal{M} = \bigcup_{\sigma^h \in \mathcal{T}^h} \overline{\sigma^h}$. Note, that the function Φ was only needed to compute $\vec{n}_\Phi$. Normal vectors to the surface can now be easily computed for each triangle σ^h. For a point $\vec{p}$ on a curve $\Gamma^m \subset \mathcal{M}$, we first need to assign $\vec{p}$ to a triangle $\sigma^h \in \mathcal{T}^h$ in which the node lies, to compute $\vec{n}_\Phi(\vec{p})$. Further, the color data u_0 is often piecewise constant and uniquely given by its value on the triangles. To evaluate $u_0(\vec{p})$, we also need to assign the node to its corresponding triangle.

For each simplex, we can project an element of $\mathbb{R}^3$ to the simplex plane and can use barycentric coordinates to determine if the projected node lies inside the triangle. Surfaces are often composed of 10^5 to 10^6 triangles. Therefore, for a given point, finding the corresponding triangle in which the point lies results in a high computational effort if no prior knowledge is used.

For $m = 0$ and a curve Γ_i^m, $i \in \{1, \dots, N_C\}$, with nodes $\vec{X}_{i,j}^m$, $j = j_0^i, \dots, N_i$, we perform a global search only for $\vec{X}_{i,j_0^i}^m$. For $j > j_0^i$, we consider first the simplex to which $\vec{X}_{i,j-1}^m$ has been assigned. If the node $\vec{X}_{i,j}^m$ is not located in the same simplex, we start a search considering

successively the neighbor simplices. For $m > 0$, we can assume that a node has moved only slightly on the surface from step $m-1$ to m. Therefore, we start the search using the triangle to which the node was assigned in time step $m-1$. Consequently, a global search has to be performed only N_C times at the beginning of the segmentation.

After the linear system (4.41b) has been solved, some of the nodes may not lie exactly on the surface. For smooth surfaces (like spheres, tori, etc.), the nodes stay very close to the surface if small time steps are used, see Barrett et al. (2010a). However, for triangulated surfaces, a reprojection onto the surface is necessary since $\vec{\nu}_\Phi$ is not continuous. The projection onto the surface does not result in an additional computational effort: In the next time step, we need to compute $\vec{\nu}_\Phi = \vec{n}_\Phi \circ \vec{x}$ and need to evaluate u_0 again for each node. For both, we have to determine again the closest triangle for a point. As described above, this is done by projection of the original node to the triangle plane and by using barycentric coordinates. I.e. we already need to determine a projection of the original point to the corresponding triangle.

Computation of regions and coefficients

For the external forcing term, we need to determine the regions Ω_k^m, approximations of $\Omega_k(t_m)$, $k = 1, \ldots, N_R$. The regions Ω_k^m are separated and thus determined by the union of discrete curves $\Gamma^m = \Gamma_1^m \cup \ldots \cup \Gamma_{N_C}^m$. Further, we need to compute the coefficients c_k^m which are the average color values of the image function u_0 on the corresponding regions or average values of components with respect to the CB or HSV color space.

For $m = 0$, we need to assign each simplex $\sigma^h \in \mathcal{T}^h$ to a region Ω_k^0. For $m > 0$, we need to update the assignment only in a neighborhood of the curves.

Let $\vec{p}_{\sigma^h,j}$, $j = 1, 2, 3$, denote the vertices of a triangle σ^h. We assign the simplex to a region Ω_k^m if its center $\vec{p}_{\sigma^h} = (\vec{p}_{\sigma^h,1} + \vec{p}_{\sigma^h,2} + \vec{p}_{\sigma^h,3})/3$ belongs to the region. In the rare case, that $\vec{p}_{\sigma^h}$ lies directly on a curve Γ_i^m, it is assigned either to $\Omega_{k^+(i)}^m$ or $\Omega_{k^-(i)}^m$. For image segmentation, we do not apply any special treatment to simplices which are truncated by a curve.

For a simplex σ^h close to a curve with center $\vec{p}_{\sigma^h}$, we can search for the closest node $\vec{X}_{i,j}^m$ and consider the sign of $(\vec{p}_{\sigma^h} - \vec{X}_{i,j}^m) \,.\, \vec{\omega}_{\mathcal{M}}^m(q_j^i)$.

For $m = 0$, we also need to consider simplices which are not close to a curve. The direction $\vec{\omega}_{\mathcal{M}}^m(q_j^i)$ cannot be used for remote simplices if the surface $\mathcal{M}$ is curved. Having assigned a small band of simplices around the curves, the remaining simplices can inherit the region index by using the neighbor relation between the simplices of the triangulation.

Motivated by these thoughts, we propose the following algorithm for computation of the regions:

- For all nodes $\vec{X}_{i,j}^m$, $i = 1, \ldots, N_C$, $j = j_0^i, \ldots, N_i$, we consider the triangle σ^h to which $\vec{X}_{i,j}^m$ belongs (see determination of the closest triangle described above). If $(\vec{p}_{\sigma^h} - \vec{X}_{i,j}^m) \,.\, \vec{\omega}_{\mathcal{M}}^m(q_j^i)$ is positive, the simplex is assigned to $\Omega_{k^+(i)}^m$, otherwise to $\Omega_{k^-(i)}^m$. The indices of neighbor simplices of σ^h are stored in a list.
- We consider the simplices of the auxiliary list, which have not been assigned to a region yet. For a simplex σ^h of the list, we search for the closest node point $\vec{X}_{i,j}^m$ and determine a region index using the sign of $(\vec{p}_{\sigma^h} - \vec{X}_{i,j}^m) \,.\, \vec{\omega}_{\mathcal{M}}^m(q_j^i)$. We store the neighbor simplices of σ^h in a new list.

Figure 4.3: Illustration how triangles are assigned to a region. Left: Image and initial curve. Center: Small band after $n_0 = 4$ steps. Right: Regions colored with mean brightness value after assignment of all triangles. The surface data is from the Stanford Computer Graphics Laboratory, cf. Turk and Levoy (1994).

- Having assigned all simplices of the current list to a region, the current list is deleted and the simplices of the new list are considered. We repeat the procedure n_0 times. Afterwards, a small band of simplices around each curve is assigned to regions.
- For $m = 0$, the remaining simplices are considered successively by using again lists of neighbor simplices. After the step n_0, we do not determine the closest node of a curve. A new simplex inherits the region index directly from its neighbor.

Figure 4.3 illustrates the assignment of regions for simplices of a triangulated surface. It shows the Stanford Bunny from the Stanford Computer Graphics Laboratory (`https://graphics.stanford.edu/data/3Dscanrep/`), cf. Turk and Levoy (1994), and an image with three small discs painted on its surface. We used $n_0 = 4$ levels of neighbor simplices to assign the simplices of a small band around the initial curve to one of the two regions separated by the initial curve. The remaining simplices (marked with dark color in the second subfigure) are finally assigned to a region by heritage of the region index.

The final coefficients are computed as for planar images, cf. Section 3.3.6: Let $C_k^m = \sum_{\sigma^h \in \Omega_k^m} u_0|_{\sigma^h}$ denote the sum of the color data and n_k^m the number of simplices belonging to Ω_k^m. The coefficient for the Chan-Vese model is computed by setting $c_k^m = C_k^m / n_k^m$. For color spaces like the CB or HSV space, we need to make use of normalized means for some components of the color, cf. Section 3.3.6.

For $m > 0$, we need to update the coefficients only close the curve, i.e. we consider only the simplices of a small band around the curves (using again n_0 levels of neighbor simplices around the curves). If a simplex σ^h changes its region assignment from Ω_l^m to Ω_k^m, we set

$$n_k^m = n_k^m + 1, \quad n_l^m = n_l^m - 1, \quad C_k^m = C_k^m + u_0|_{\sigma^h}, \quad C_l^m = C_l^m - u_0|_{\sigma^h}. \tag{4.46}$$

Global refinement-coarsening strategy

We perform a global refinement-coarsening similar as for the planar case by using two thresholds $l_{\max}$ and $l_{\min}$ for the average distance between neighbor nodes, see Section 3.3.6. Let N_i^m be the number of nodes belonging to a curve Γ_i^m. If $|\Gamma_i^m| / N_i^m > l_{\max}$, we perform a global refinement of the curve by inserting a new node between two neighbor nodes. On the contrary, if $|\Gamma_i^m| / N_i^m < l_{\min}$, we perform a global coarsening, i.e. we delete every second node of the polygonal curve which is not a boundary point.

When inserting a new node between two nodes $\vec{X}^m_{i,j}$ and $\vec{X}^m_{i,j+1}$ during a refinement, we first compute $\vec{p} = (\vec{X}^m_{i,j} + \vec{X}^m_{i,j+1})/2$ and determine the closest triangle (again the search for the closest triangle is done efficiently by starting with the triangle in which e.g. $\vec{X}^m_{i,j}$ lies). Having found the closest triangle σ^h, $\vec{p}$ is projected orthogonally to σ^h. Consequently, all newly generated nodes lie on the surface.

4.3.6 Numerical Solution of the Image Restoration Scheme

In this section, we describe how the scheme (4.27) for image restoration can be solved numerically with a finite element approach. As above, we consider a polyhedral surface $\mathcal{M}$ given by a set $\mathcal{T}^h$ of triangles.

The image denoising is performed as postprocessing step using the final regions from the time step $m = M$. The surface thus consists of polyhedral regions $\Omega^h_k := \Omega^M_k$, $k = 1, \dots, N_R$. Let $\mathcal{T}^h_k = \{\sigma^h \in \mathcal{T}^h : \sigma^h \subset \Omega^h_k\}$ denote the set of triangles belonging to Ω^h_k and let $\vec{p}_{k,j}$, $j = 1, \dots, N^h_k$, denote the vertices of the triangles belonging to $\mathcal{T}^h_k$.

For each $k = 1, \dots, N_R$, we define the following finite element space

$$S^h_k := \left\{ u^h \in C(\overline{\Omega^h_k}, \mathbb{R}) : u^h{}_{|\sigma^h} \text{ is linear } \forall \sigma^h \in \mathcal{T}^h_k \right\}. \tag{4.47}$$

For piecewise continuous functions $u^h, v^h : \Omega^h_k \to \mathbb{R}^{(3)}$ with possible jumps at edges of simplices $\sigma^h \in \mathcal{T}^h_k$, we define the mass lumped inner product

$$\langle u^h, v^h \rangle^h := \frac{1}{3} \sum_{\sigma^h \in \mathcal{T}^h_k} |\sigma^h| \sum_{j=1}^{3} (u^h \,.\, v^h)((\vec{p}_{\sigma^h,j})^-), \tag{4.48}$$

where $|\sigma^h|$ denotes the area of σ^h, and as above $\vec{p}_{\sigma^h,j}$, $j = 1, 2, 3$, denote the vertices of the triangle $\sigma^h \in \mathcal{T}^h_k$ and

$$u^h((\vec{p}_{\sigma^h,j})^-) := \lim_{\vec{p} \to \vec{p}_{\sigma^h,j}, \vec{p} \in \sigma^h} u^h(\vec{p}).$$

Further, for functions $u^h, v^h \in L^2(\Omega^h_k, \mathbb{R}^{(3)})$, we define

$$\langle u^h, v^h \rangle := \int_{\Omega^h_k} u^h \,.\, v^h \,\mathrm{d}A. \tag{4.49}$$

We consider the following discrete system for each region $k \in \{1, \dots, N_R\}$: Find $u^h \in S^h_k$ such that

$$\frac{1}{\lambda} \langle \nabla_{\mathcal{M}} u^h, \nabla_{\mathcal{M}} v^h \rangle + \langle u^h, v^h \rangle^h = \langle u_0, v^h \rangle^h, \quad \forall v^h \in S^h_k, \tag{4.50}$$

where $\lambda > 0$ is a weighting parameter (cf. (4.8)).

Let $\{\phi^h_{k,i}\}_{i=1}^{N^h_k}$ with $\phi^h_{k,i}(\vec{p}_{k,j}) = \delta_{ij}$ denote the standard basis of S^h_k. Using this standard basis, we can identify each element in S^h_k with its coefficient vector in $\mathbb{R}^{N^h_k}$. Further, we define the matrices $M^h_k, A^h_k \in \mathbb{R}^{N^h_k \times N^h_k}$ by

$$(M^h_k)_{ij} := \langle \phi^h_{k,i}, \phi^h_{k,j} \rangle^h, \quad (A^h_k)_{ij} := \langle \nabla_{\mathcal{M}} \phi^h_{k,i}, \nabla_{\mathcal{M}} \phi^h_{k,j} \rangle, \quad i, j = 1, \dots, N^h_k. \tag{4.51}$$

The entries of the matrices M^h_k and A^h_k are computed by considering each triangle $\sigma^h \in \mathcal{T}^h_k$ and computing the contribution of σ^h to the entries corresponding to the indices of its vertices. For computing the contribution of σ^h to A^h_k we need to compute surface gradients.

For that, we consider three nodes $\vec{p}_{k,j_1}, \vec{p}_{k,j_2}$ and $\vec{p}_{k,j_3}$, $j_1, j_2, j_3 \in \{1, \dots, N_k^h\}$, being the vertices of a triangle σ^h in $\mathcal{T}_k^h$. For the ease of notation, we assume $j_1 = 1$, $j_2 = 2$ and $j_3 = 3$. One tangential vector is given by

$$\vec{\tau}_1 := \frac{\vec{p}_{k,3} - \vec{p}_{k,2}}{\|\vec{p}_{k,3} - \vec{p}_{k,2}\|}.$$

A second tangential vector which is orthogonal to $\vec{\tau}_1$ can be obtained by

$$\vec{\tau}_2 := \frac{\vec{p}_{k,1} - \vec{q}_k}{\|\vec{p}_{k,1} - \vec{q}_k\|},$$

with

$$\vec{q}_k = \vec{p}_{k,2} + ((\vec{p}_{k,1} - \vec{p}_{k,2}) \cdot \vec{\tau}_1)\, \vec{\tau}_1.$$

The vector $\vec{\tau}_2$ is the vector in the tangential plane which can also be obtained by a positive 90°-rotation of $\vec{\tau}_1$ around the axis given by the normal vector to the triangle.

We note that $\partial_{\vec{\tau}_1} \phi_{k,1}^h = 0$. For $\partial_{\vec{\tau}_2} \phi_{k,1}^h$ we consider a curve $\gamma : [0, \|\vec{p}_{k,1} - \vec{q}_k\|] \to \mathbb{R}^3$, $\gamma(\epsilon) = \vec{q}_k + \epsilon\, \vec{\tau}_2$. The composition $\phi_{k,1}^h \circ \gamma$ is given by

$$(\phi_{k,1}^h \circ \gamma)(\epsilon) = \frac{\epsilon}{\|\vec{p}_{k,1} - \vec{q}_k\|}.$$

Using the definition of the directional derivative (2.11), we get

$$\partial_{\vec{\tau}_2} \phi_{k,1}^h = \frac{\mathrm{d}}{\mathrm{d}\epsilon} \phi_{k,1}(\gamma(\epsilon))|_{\epsilon=0} = \frac{1}{\|\vec{p}_{k,1} - \vec{q}_k\|}.$$

The surface gradient is then given by

$$\nabla_{\mathcal{M}} \phi_{k,1}^h|_{\sigma^h} = \partial_{\vec{\tau}_1} \phi_{k,1}^h\, \vec{\tau}_1 + \partial_{\vec{\tau}_2} \phi_{k,1}^h\, \vec{\tau}_2 = \frac{\vec{p}_{k,1} - \vec{q}_k}{\|\vec{p}_{k,1} - \vec{q}_k\|^2}.$$

Similarly, we compute $\nabla_{\mathcal{M}} \phi_{k,2}^h|_{\sigma^h}$ and $\nabla_{\mathcal{M}} \phi_{k,3}^h|_{\sigma^h}$.

The discrete equation (4.50) can be rewritten to the following linear system: Find $u^h \in \mathbb{R}^{N_k^h}$ such that

$$\frac{1}{\lambda} A_k^h u^h + M_k^h u^h = M_k^h U_0, \tag{4.52}$$

holds, where $U_0 \in \mathbb{R}^{N_k^h}$ is given by

$$(U_0)_j = \frac{\sum_{\sigma^h \in \mathcal{T}_{k,j}^h} |\sigma^h|\, (u_0)_{|\sigma^h}}{\sum_{\sigma^h \in \mathcal{T}_{k,j}^h} |\sigma^h|}, \quad j = 1, \dots, N_k^h, \tag{4.53}$$

with

$$\mathcal{T}_{k,j}^h = \left\{ \sigma^h \in \mathcal{T}_k^h \,:\, \vec{p}_{k,j} \in \overline{\sigma^h} \right\}.$$

Using this definition of U_0, we obtain $\langle u_0, \phi_{k,j}^h \rangle^h = (M_k^h U_0)_j$ for $j = 1, \dots, N_k^h$.

Note, that the Neumann boundary conditions are automatically incorporated in the scheme (4.52). The finite element approach is based on a weak formulation of (4.27) which contains the Neumann boundary conditions as natural conditions.

We obtain an element-wise constant image approximation U^h by setting

$$U^h{}_{|\sigma^h} = \frac{1}{3} \sum_{j=1}^{3} u^h(\vec{p}_{\sigma^h,j}). \tag{4.54}$$

for simplices $\sigma^h \in \mathcal{T}_k^h$. By solving the diffusion equation on each region independently, the boundaries of the regions are not smoothed out. Vector-valued images are denoised by applying the method on each component.

4.3.7 Summary of the Image Processing Algorithm

We propose the following algorithm for image segmentation with postprocessing image restoration: Given a set of polygonal curves $\Gamma^0 = (\Gamma_1^0, \ldots, \Gamma_{N_C}^0)$ and $\vec{X}^0 = (\vec{X}_1^0, \ldots, \vec{X}_{N_C}^0)$ with $\vec{X}_i^0(I_i) = \Gamma_i^0$, $\vec{X}_i^0(q_j^i) \in \mathcal{M}$, $i = 1, \ldots, N_C$, $j = j_0^i, \ldots, N_i$, we perform the following steps for $m = 0, 1, \ldots, M-1$:

(i) Compute the regions $\Omega_k^m \subset \mathcal{M}$ and the coefficients c_k^m, $k = 1, \ldots, N_R$, as described in Section 4.3.5.

(ii) Compute b^m as defined in (4.39) by using the coefficients c_k^m of step (i). Compute $\vec{X}^{m+1} = \vec{X}^m + \delta\vec{X}^{m+1}$ by solving the linear equation (4.41b), see Section 4.3.2.

(iii) Check if topology changes occur, see Section 4.3.4. In case of a topology change, except for a pure deletion of a curve, repeat the steps (i) and (ii) n_{sub}-times with a step size of τ_m/n_{sub} and execute the topology change when it occurs in a substep.

(iv) If necessary, perform global coarsening or refinement as described in Section 4.3.5.

Having found a final segmentation of the image at time $m = M$, perform the image denoising by computing a piecewise smooth approximation of the image function as presented in Section 4.3.6.

In principle, we can also use adaptive time step sizes and adaptive values for the weighting parameter σ, similar as in the planar case, cf. Section 3.3.6.

4.4 Results

4.4.1 Artificial Test Images

Images on the Stanford bunny

We test the developed algorithm for image segmentation by considering artificial images on the Standford bunny, from the Stanford Computer Graphics Laboratory (`https://graphics.stanford.edu/data/3Dscanrep/`), cf. Turk and Levoy (1994). As a first example, we consider a gray-scaled image showing three dark discs. This example is similar to an experiment presented in Krüger et al. (2008), who apply a level set method based on the work of Cheng et al. (2002), where the intersection of two zero level sets are used to describe a curve on a surface. Further, Krüger et al. (2008) use a geodesic active contours model (recall (4.7)) with a balloon force, whereas we apply the Chan-Vese model extended to images on surfaces.

Figure 4.4 shows our image segmentation result using the developed direct, parametric approach for image segmentation. We use the parameters $\sigma = 2$ and $\lambda = 50$ to weight the curvature and external term. Let $\Delta t = \tau_m$ denote the time step size. The time step size is set to $\Delta t = 0.01$. This example demonstrates topology changes; in detail it shows how one initial closed curve is split up in three single curves. The contours at four different time steps and the corresponding piecewise constant approximation are presented in Figure 4.4. Of course, a level set technique as used by Krüger et al. (2008) can handle splitting automatically, whereas we need to detect the change in topology explicitly using the method described in Section 4.3.4. However, the method to detect topology changes is efficient, since

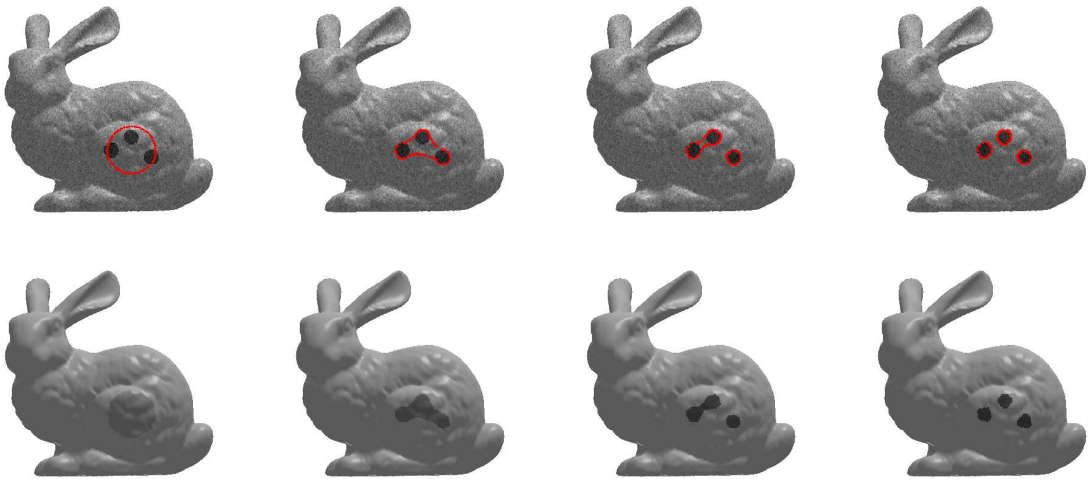


Figure 4.4: Image segmentation of a gray-scaled image on the Stanford bunny. Demonstration of topology changes (splitting). First row: Original image and contours for $m = 1, 100, 140, 150$. Second row: Piecewise constant approximation. The surface is from the Stanford Computer Graphics Laboratory, cf. Turk and Levoy (1994).

it has a computational effort of $\mathcal{O}(N)$, where N is the number of node points of the polygonal curves.

In a second experiment, we demonstrate the creation and handling of triple junctions for a curve network on the Stanford bunny, cf. Figure 4.5. For this experiment, we set $\sigma = 1$, $\lambda = 40$ and $\Delta t = 0.01$. The possibility of triple junctions is not considered in Krüger et al. (2008).

Further, we apply the image denoising method described in Section 4.2.5 and 4.3.6. Since the term $\Delta_{\mathcal{M}} u_k$ in (4.27) is weighted with $1/\lambda$, the denoised image is close to the original image, the larger λ is chosen, cf. Figure 4.6. Setting $\lambda = 1000$ or $\lambda = 10000$, the noise is not completely smoothed out. Setting $\lambda = 0.1$ the resulting denoised version is close to a piecewise constant image approximation. Setting $\lambda = 100$ results in a good denoised image.

Image on a torus

We consider another artificial image painted on a torus to demonstrate that the method can be applied on surfaces of arbitrary topology type (for example arbitrary genus). Figure 4.7 shows a torus with a color image from different viewing angles and the curves at different iteration steps during the evolution. The RGB color space is used for the segmentation, cf. Section 3.2.9. As weighting parameters $\sigma = 1$ and $\lambda_1 = \lambda_2 = \lambda_3 = 20$ are used (i.e. the three components of the color image data are weighted equally). The time step size was set to $\Delta t = 0.0001$. Figure 4.8 shows the piecewise constant approximation for each iteration step.

Two different topology changes occur in this example: Around $m = 335$, two triple junctions and a new curve are created. Shortly after $m = 422$, one blue curve splits into two single curves.

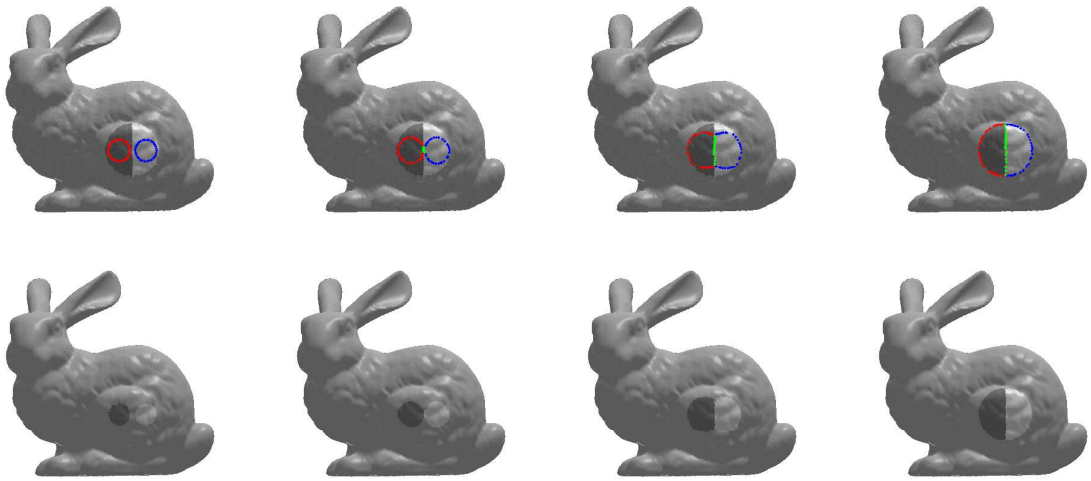


Figure 4.5: Image segmentation of a gray-scaled image on the Stanford bunny. Curve network with triple junctions. First row: Original image and contours for $m = 1, 42, 100, 250$. Second row: Piecewise constant approximation. The surface is from the Stanford Computer Graphics Laboratory, cf. Turk and Levoy (1994).

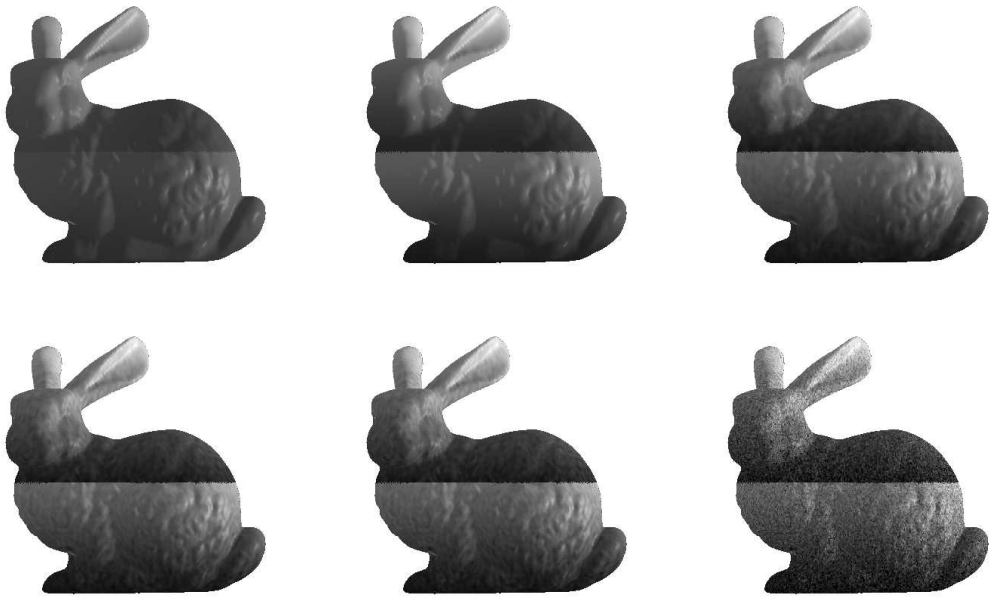


Figure 4.6: Image denoising with edge enhancement. Subfigure 1-5 (row-wise): Denoising result using $\lambda = 0.1, 1, 100, 1000, 10000$. Subfigure 6: Original image with noise. The surface is from the Stanford Computer Graphics Laboratory, cf. Turk and Levoy (1994).

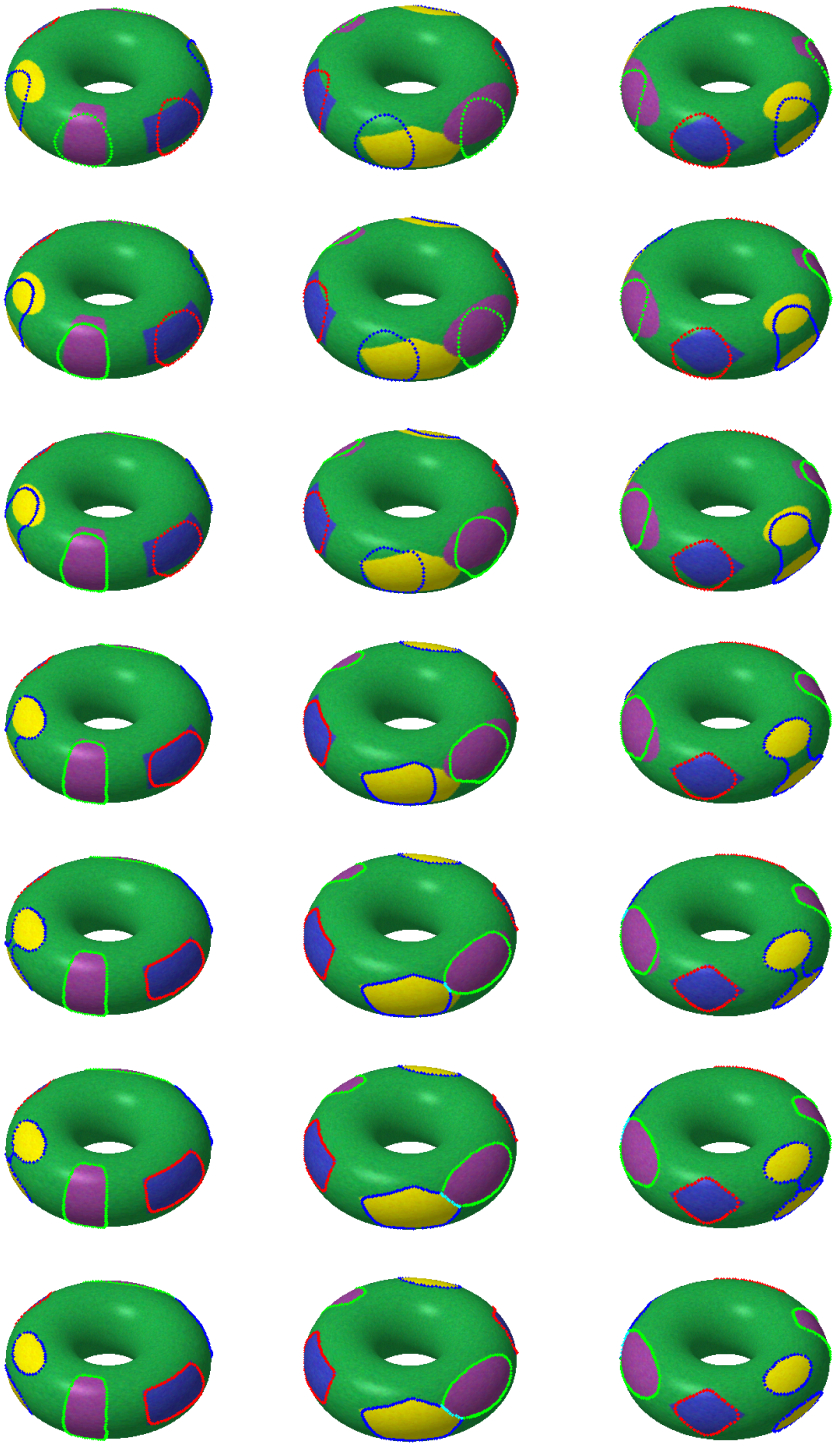


Figure 4.7: Segmentation of an artificial image showing different objects on a torus. First - seventh row: Original image and contours for $m = 1, 50, 100, 200, 335, 422, 600$. First - third column: Different viewing angles.

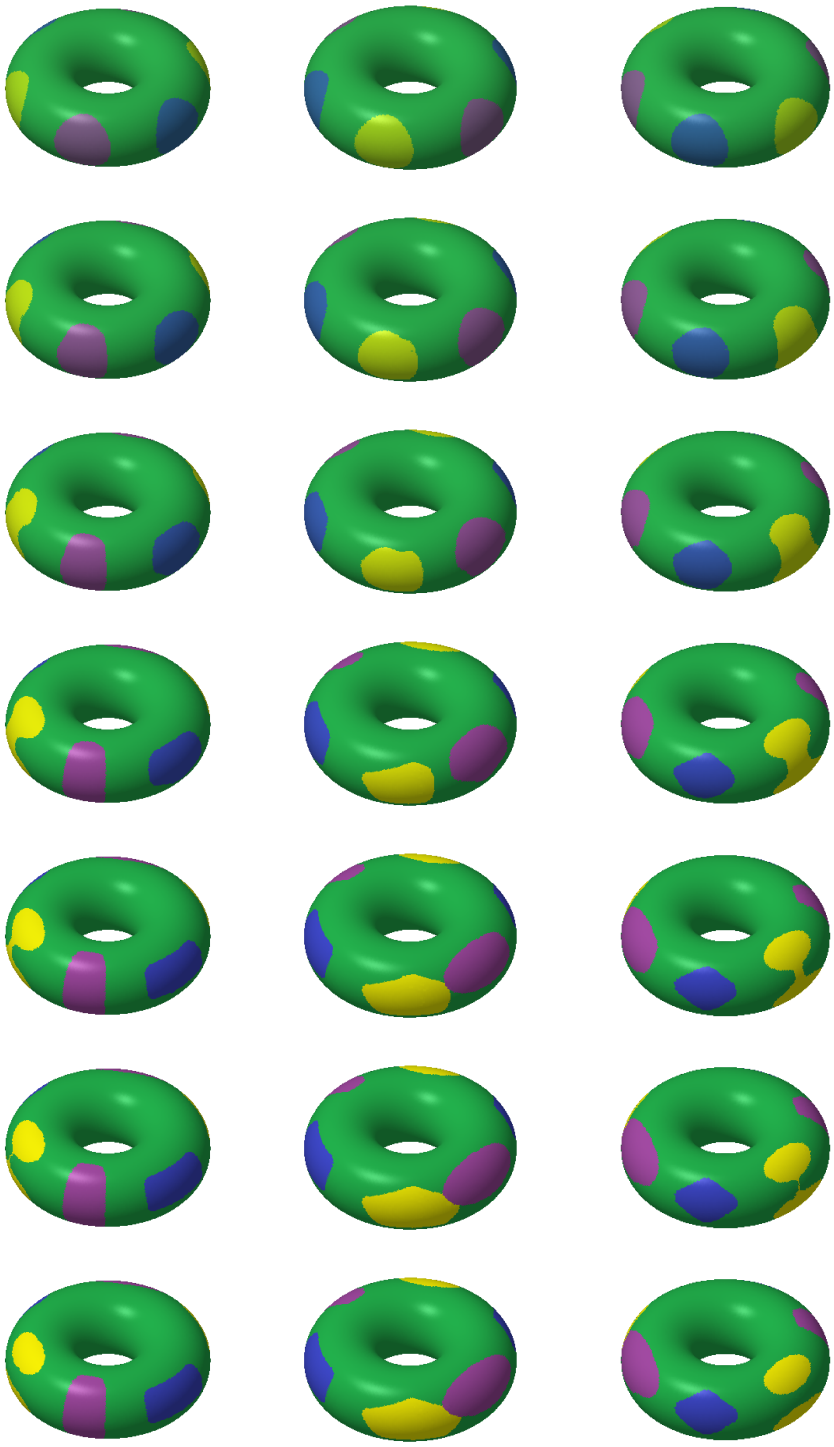


Figure 4.8: Segmentation of an artificial image showing different objects on a torus. First - seventh row: Piecewise constant approximation for for $m = 1, 50, 100, 200, 335, 422, 600$. First - third column: Different viewing angles.

4.4.2 Real Images

Lip contour segmentation

We consider an application where lip contours should be detected on given face image data. Figure 4.9 shows the results where the image segmentation algorithm is applied on three sample images of the 3D face scans from the 3D Basel Face Model (BFM) published by the Computer Science department of the University of Basel (`http://faces.cs.unibas.ch/bfm/main.php?nav=1-0&id=basel_face_model`), see also Paysan et al. (2009). The images are segmented using the chromaticity-brightness color space, cf. Section 3.2.9. We used $\sigma = 0.25$, $\lambda_C = 200$, $\lambda_B = 20$ and $\Delta t = 0.001$. The initial contour is a closed curve placed around the lips. Applying our algorithm for image segmentation, we obtain the final lip contours.

Processing of global Earth observation data

Another application is the processing of Earth observation data. Global Earth observation data can be interpreted as an image given on a sphere. We apply the image segmentation and denoising method on data from the NASA Earth Observation data set (`http://neo.sci.gsfc.nasa.gov/`), cf. NASA (2014). We segment an image showing outgoing longwave radiation[1], see Figure 4.10 - 4.12. The colors represent the amount of outgoing longwave radiation leaving the Earth's atmosphere in one month (here: January 2014). Yellow and orange color represent greater heat emission (around 300 to 350 Wm^{-2}); purple and blue color represent intermediate emissions (around 200 Wm^{-2}).

Figure 4.10 presents the original image with the contours at different time steps (rows), observed from different viewing angles (columns). For the segmentation we used the RGB color space and set the weighting parameters for the curvature and the external term to $\sigma = 1$ and $\lambda_1 = \lambda_2 = \lambda_3 = \lambda = 50$. As time step size we set $\Delta t = 0.0005$. Several topology changes (splitting and merging) occur during the segmentation (cf. e.g. $m = 75$). Figure 4.11 shows the corresponding piecewise constant approximations. Figure 4.12 presents the result of the postprocessing image restoration using the parameters $\lambda = 100$, $\lambda = 1000$ and $\lambda = 10000$.

For demonstration of multiphase image segmentation, we consider a second example image from the NASA Earth Observation data set (`http://neo.sci.gsfc.nasa.gov/`), cf. NASA (2014). We now segment an image showing the Earth's net radiation. The net radiation is defined as the difference between the amount of solar energy which enters the Earth system and the amount of heat energy which escapes into space during one month (here: March 2014). Red color represents a net radiation around 280 Wm^{-2}, yellow color a net radiation around 0 Wm^{-2} and blue-green color a net radiation of -280 Wm^{-2}.

Figure 4.13 presents the original image with the contours at different time steps (rows), observed from different viewing angles (columns). For the segmentation we used $\sigma = 1$ and $\lambda_1 = \lambda_2 = \lambda_3 = \lambda = 300$ and the RGB color space. As time step size we set $\Delta t = 0.002$. The detected regions are not separated by sharp image edges. Here, the detected boundaries are weak edges, i.e. they lie at locations in the image where the color smoothly changes from yellow to orange or yellow to green, respectively. At $m = 71$ a merging and at $m = 149$ a splitting occurs. Figure 4.14 shows the corresponding piecewise constant approximation.

[1]Imagery by Jesse Allen, NASA Earth Observatory, based on FLASHFlux data. FLASHFlux data are produced using CERES observations convolved with MODIS measurements from both the Terra and Aqua satellite. Data provided by the FLASHFlux team, NASA Langley Research Center.

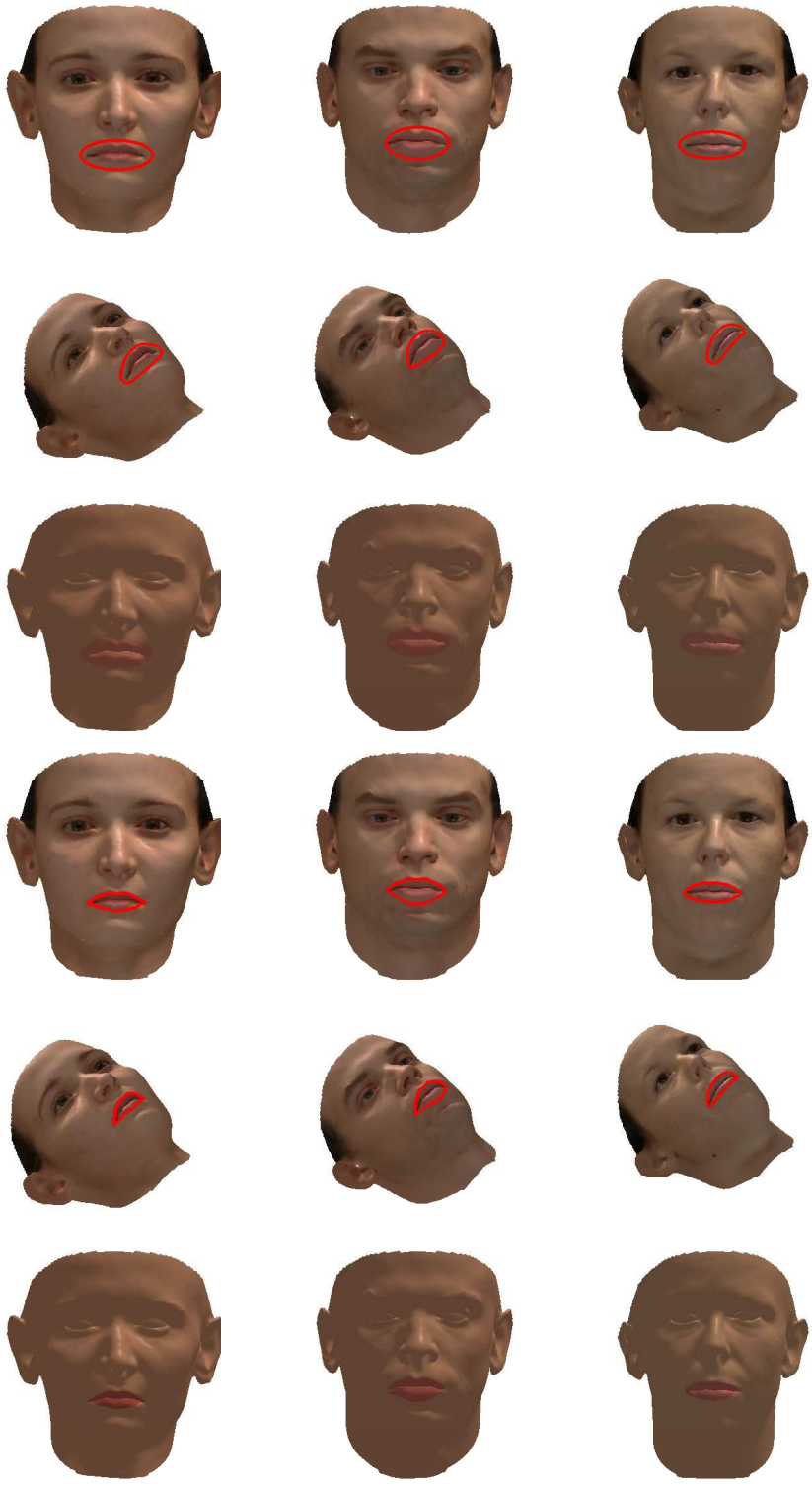


Figure 4.9: Lip contour detection. First and second row: Initial contours (two different viewing angles). Third row: Initial piecewise constant approximation. Fourth and fifth row: Final contours (two different viewing angles). Sixth row: Final piecewise constant approximation. The surfaces and images are from the 3D Basel Face Model (BFM) of the Computer Science department of the University of Basel, cf. Paysan et al. (2009).

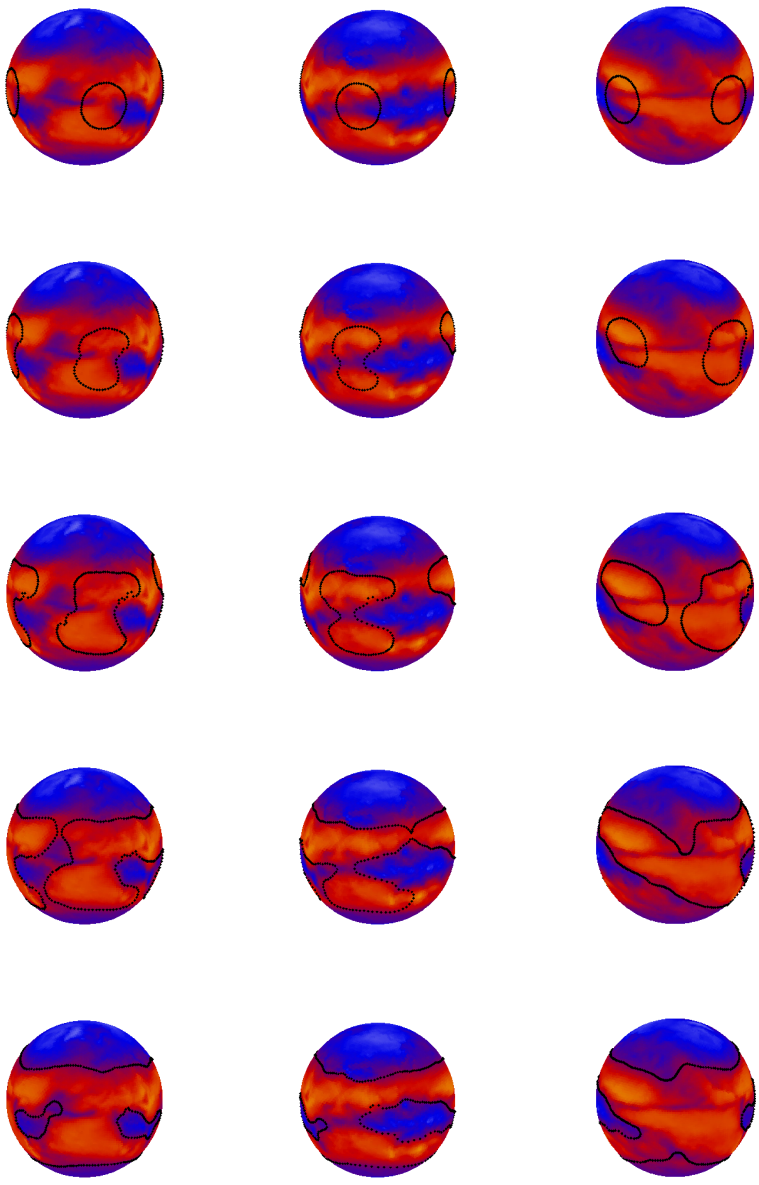


Figure 4.10: Segmentation of longwave radiation data given on the Earth's surface. First - fifth row: Original image and contours for $m = 1, 20, 50, 75, 150$. First - third column: Different viewing angles. The original image is from the NASA Earth Observation data set, NASA (2014).

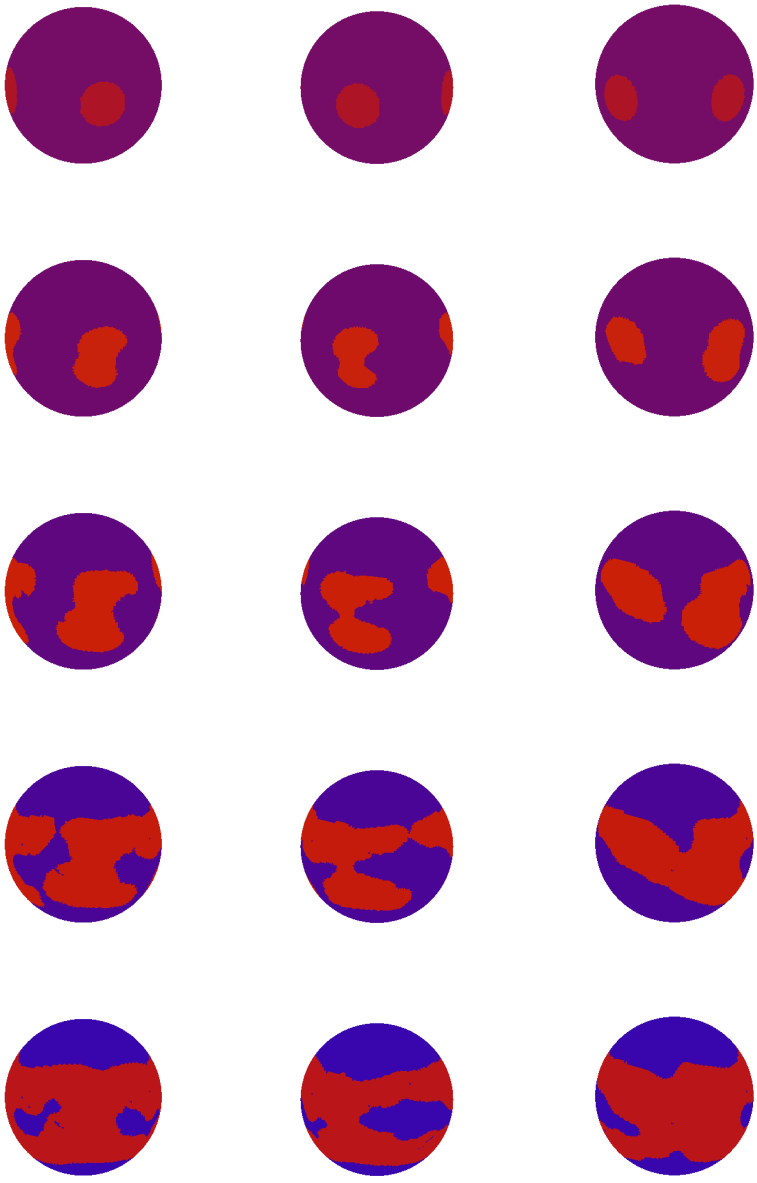


Figure 4.11: Segmentation of longwave radiation data given on the Earth's surface. First - fifth row: Piecewise constant approximation for $m = 1, 20, 50, 75, 150$. First - third column: Different viewing angles. The original image is from the NASA Earth Observation data set, NASA (2014).

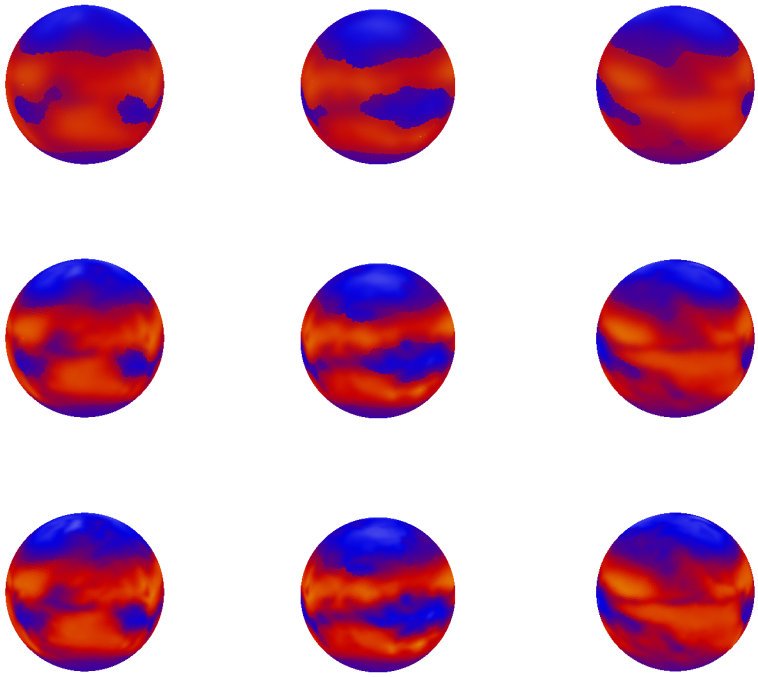


Figure 4.12: Denoising of the longwave radiation data using the detected regions from time step $m = M = 150$. First row: $\lambda = 100$. Second row: $\lambda = 1000$. Third row: $\lambda = 10000$. First - third column: Different viewing angles. The original image is from the NASA Earth Observation data set, NASA (2014).

Figure 4.15 presents the result of the postprocessing image denoising using the parameters $\lambda = 100$, $\lambda = 1000$ and $\lambda = 10000$.

Since the original image has little noise, we added some artificial, Gaussian noise to the image and repeated the segmentation using the generated image, see Figure 4.16. It presents the original, noise-free image (first row) and the noise added image (second row) from different viewing angles.

Figure 4.17 shows the segmentation results, i.e. the noise-added image and the final contours (first row) and the final piecewise constant approximation (second row). The image can be successfully segmented also in the presence of noise. As postprocessing step, we smooth the noisy image as described in Section 4.3.6. Figure 4.18 shows the denoised image for different parameters λ under different viewing angles. We compare the results for several choices of λ: Using $\lambda = 100$, the denoised image is very close to the piecewise constant image. For $\lambda = 1000$ and $\lambda = 10000$, we obtain images with removed noise, but which still contain sufficient details of the original data set. The data is not smoothed out too strong compared to $\lambda = 100$. Figure 4.19 shows a magnification of a part of the surface. It shows the noisy image (left) and the result of the denoising using $\lambda = 10000$ (right). We observe that the noise is well smoothed out.

We have shown how the developed method can be applied on segmentation of global Earth data. The given data need not be a classical image generated by a camera. The data can be any data defined on the Earth's surface, like radiation data as in the examples presented here.

One may argue that global Earth data can also be processed on a flat 2D domain, with a rectangular grid given by a discrete set of longitudes and latitudes. Figure 4.20 shows such a two-dimensional image with a resolution of 0.5 degrees in longitude and latitude. The original data set, see NASA (2014), contains such 2D images.

In principle, one can apply the two-dimensional image segmentation and restoration method from Chapter 3. However, performing the image processing using the 2D image has some disadvantages: To map the global Earth data to a 2D image, image boundaries at the poles and at longitude 0 have to be created. Points which are close to each other on the sphere (like left and right to longitude 0) are not close to each other in the 2D image. Also topology changes like boundary intersection will occur in the 2D image. The coordinates of the boundary nodes on the left and the right boundary of the image may not fit, i.e. they lie on longitude 0, but can have different latitude values resulting in two different 3D points. Further, the length of curves and the area of regions near the poles are differently scaled compared to those near the equator. Polar regions always appear larger in 2D images, see Figure 4.20. After a segmentation is performed, the size of the segmented regions are often of interest. Therefore, the area of the regions can be easily computed in a postprocessing step using the sphere data. This is not directly possible from the 2D image due to the different scaling of the polar and equatorial regions.

In summary, because of the above discussed facts, it is beneficial to consider global Earth data directly as images on a surface.

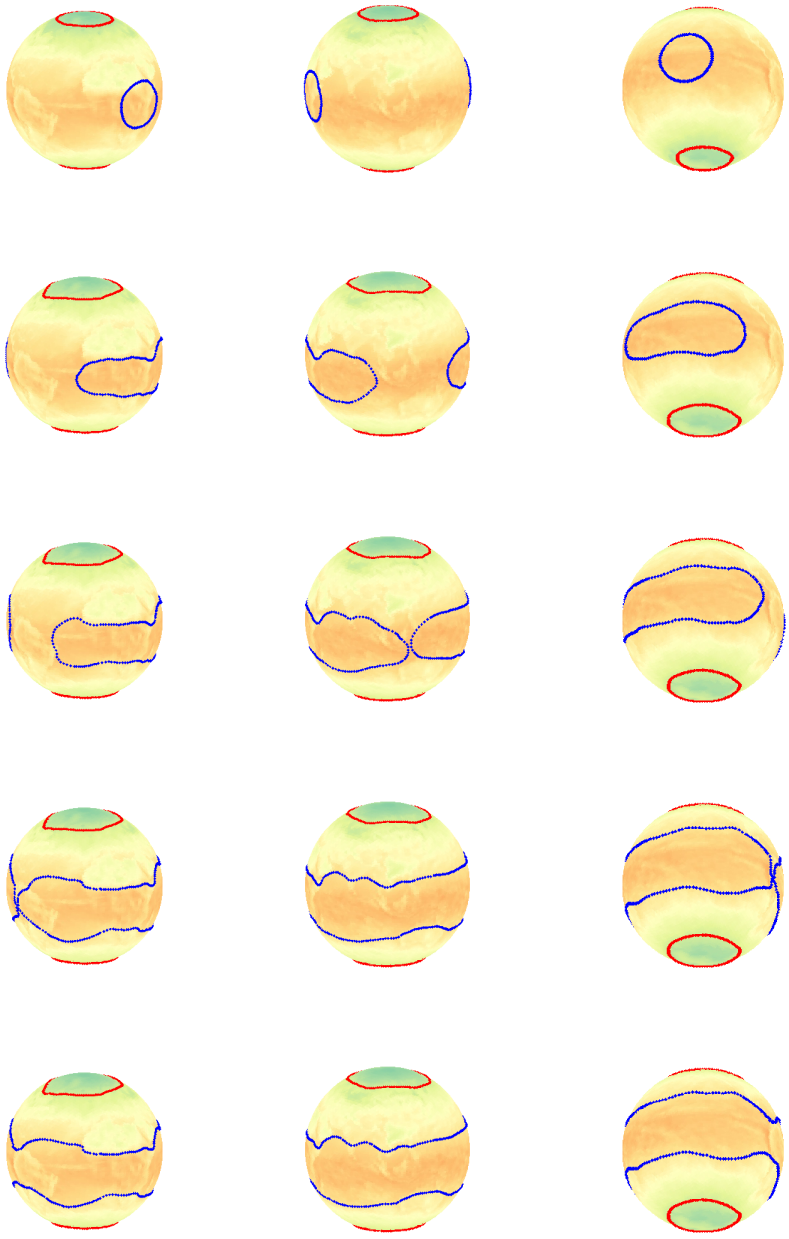


Figure 4.13: Segmentation of net radiation data given on the Earth's surface. First - fifth row: Original image and contours for $m = 1, 50, 71, 149, 180$. First - third column: Different viewing angles. The original image is from the NASA Earth Observation data set, NASA (2014).

Figure 4.14: Segmentation of net radiation data given on the Earth's surface. First - fifth row: Piecewise constant approximation for $m = 1, 50, 71, 149, 180$. First - third column: Different viewing angles. The original image is from the NASA Earth Observation data set, NASA (2014).

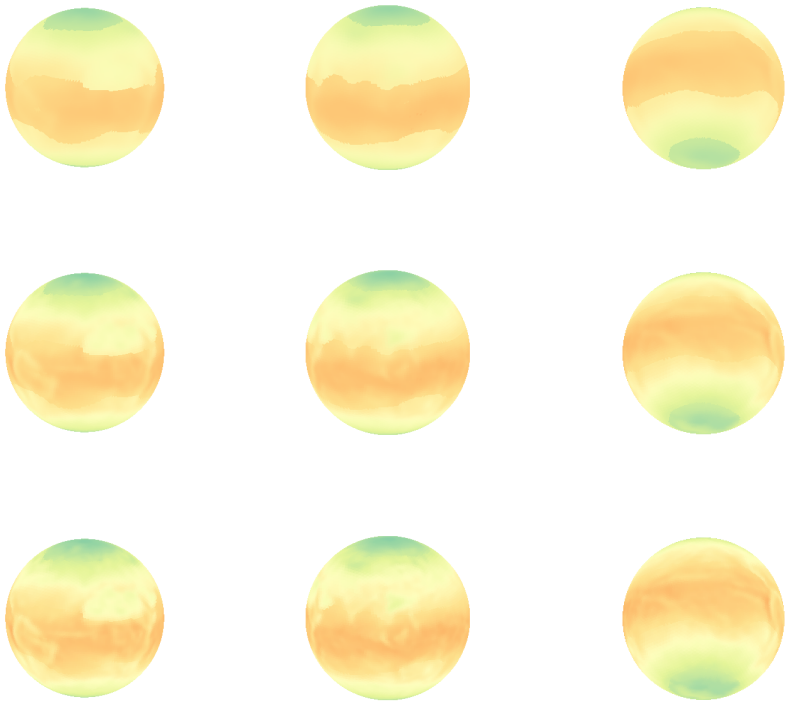


Figure 4.15: Denoising of the net radiation data using the detected regions from time step $m = M = 180$. First row: $\lambda = 100$. Second row: $\lambda = 1000$. Third row: $\lambda = 10000$. First - third column: Different viewing angles. The original image is from the NASA Earth Observation data set, NASA (2014).

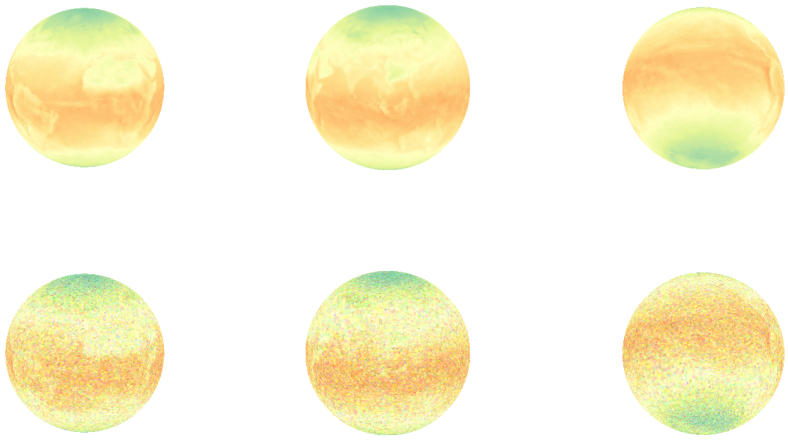


Figure 4.16: Adding noise to the net radiation data. First row: original image. Second row: noise added image. First - third column: Different viewing angles. The original image is from the NASA Earth Observation data set, NASA (2014).

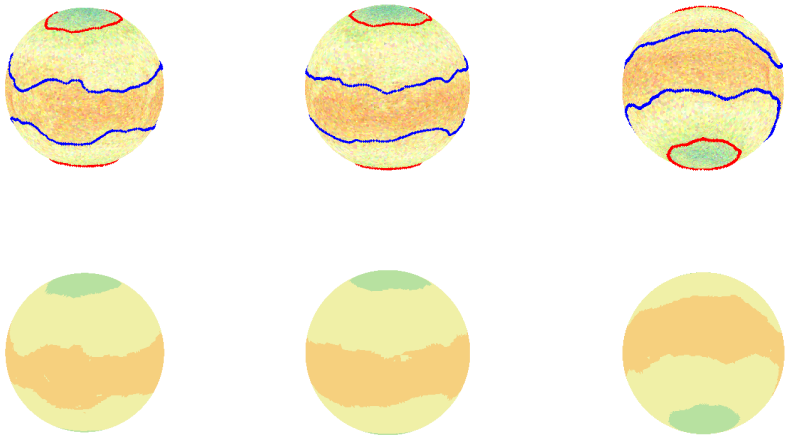


Figure 4.17: Final segmentation (first row) and piecewise constant approximation (second row) of net radiation data given on the Earth's surface with added noise, observed from different viewing angles. The original image is from the NASA Earth Observation data set, NASA (2014).

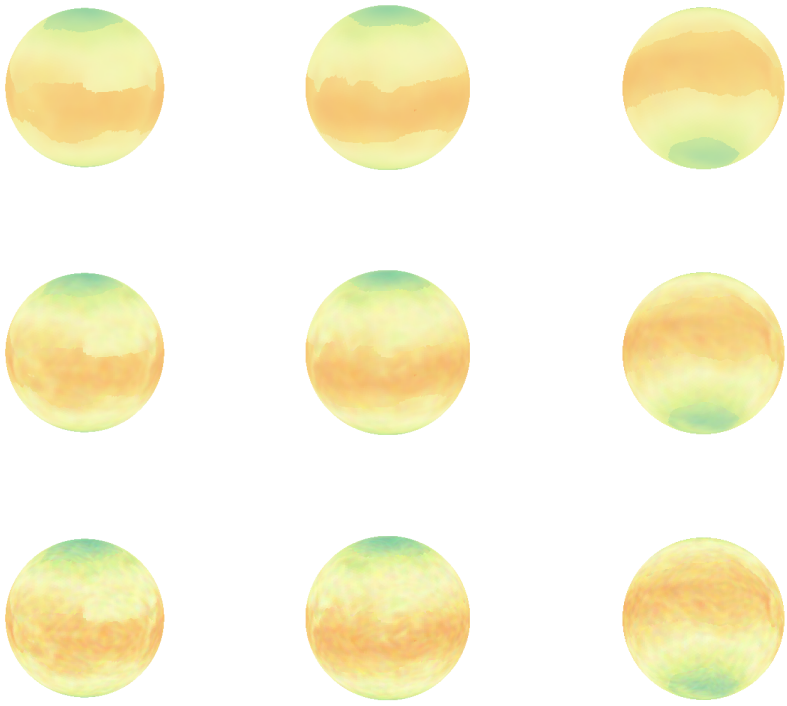


Figure 4.18: Denoising of the net radiation data (image with added noise). First row: $\lambda = 100$. Second row: $\lambda = 1000$. Third row: $\lambda = 10000$. First - third column: Different viewing angles. The original image is from the NASA Earth Observation data set, NASA (2014).

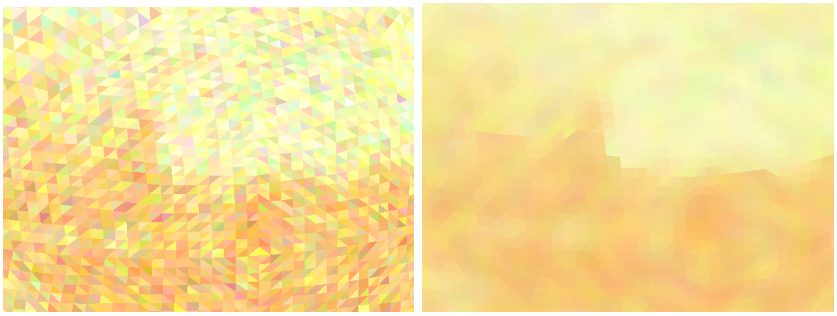


Figure 4.19: Magnification of a part of the image. Left: noisy, original image. Right: smoothed version with $\lambda = 10000$.

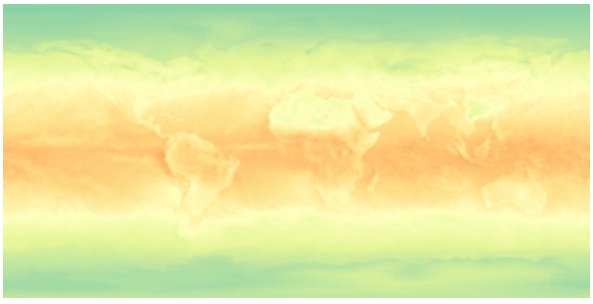


Figure 4.20: 2D image showing net radiation data. Length and area at polar and equatorial regions are not scaled equally. Therefore we process and analyze the data directly on the sphere. Source: NASA Earth Observation data set, NASA (2014).

Chapter 5

Three-dimensional Image Processing

In the previous chapters, we considered image segmentation and restoration of two-dimensional images given on planar (Chapter 3) or curved (Chapter 4) surfaces. The boundaries between the segmented regions of the images were represented by parametric curves. In the 3D case, we now consider volumetric images and the regions are three-dimensional volumes separated by two-dimensional region boundaries. The segmentation methods can be extended to 3D images by representing the boundaries by two-dimensional parametric surfaces.

5.1 Introduction

Let $\Omega \subset \mathbb{R}^3$ be open and bounded. A three-dimensional image is given by a scalar or vector-valued image function $u_0 : \Omega \to \mathbb{R}^{(d)}$. Real images are often defined on a set of $N_x \times N_y \times N_z$ voxels (=volume pixels), where u_0 is locally constant on each voxel. 3D images are often reconstructed by certain 3D imaging procedures like computer tomography (CT) or magnetic resonance imaging (MRI), cf. Udupa and Herman (1999), Scherzer et al. (2009).

3D image segmentation aims at dividing a given image in connected regions, representing 3D objects in the image or their environment in Ω. The regions are separated by a set of surfaces $\Gamma \subset \Omega$. We perform image segmentation by active surfaces, the surface-analogue to active contours. We consider time-dependent surfaces $\Gamma(t)$, $t \in [0, T]$, which evolve in time such that a certain energy functional is minimized. The Chan-Vese model (3.21), for example, can be easily extended to the 3D case and evolution equations for the surfaces $\Gamma(t)$ can be derived.

The numerical approximation requires a discretization of the evolving surfaces. In the two-dimensional case, the curves were replaced by polygonal curves given by a finite set of node points. In the three-dimensional case, the smooth surfaces are replaced by triangulated surfaces given by a finite set of triangles and a finite set of vertices.

Topology changes involving surfaces are more complex compared to topology changes involving curves. For example, if a curve splits up in two subcurves, the discretization has to be modified only at two points, recall Section 3.3.5. In detail, we had to find two close, non-neighbor nodes j and j_1 and had to connect j with $j_1 + 1$ (the former neighbor of j_1) and j_1 with $j + 1$ (the former neighbor of j). If a surface is split up in two subsurfaces, many triangles are located in a small volume. The detection of such a splitting is quite simple; the

idea of an auxiliary background grid as used for curves can be extended to topology changes of surfaces. However, the modifications of the surface triangulation are not as straight-forward as for curves. In case of splitting, we will delete the involved triangles near the splitting point resulting in two surfaces with intermediate holes. Then, we will close the intermediate holes by creating new triangles. Apart from splitting and merging, further topology changes can occur for surfaces: An increase or decrease of the genus of a surface can also occur, for example when a sphere evolves to a torus or vice versa. In summary, the execution of topology changes is the main additional challenge of 3D image segmentation with parametric surfaces.

In this chapter, we concentrate on image segmentation only. A subsequent image denoising in the segmented regions can be done similar as in the two-dimensional case, cf. Section 3.2.8 and 3.3.7. The method using finite differences can be easily transferred to the three-dimensional case. The given voxel grid can be used as spatial discretization of the volume.

5.2 Methods for Image Segmentation

5.2.1 Region-based Active Surfaces

For 3D image segmentation, the boundaries of 3D objects (regions) should be detected. The idea of active surfaces is to let a time-dependent surface, or a set of surfaces, $\Gamma(t)$, $t \in [0,T]$, evolve in time according to an evolution law designed for image segmentation such that $\Gamma(t)$ is attracted to the region boundaries in the given image.

Let $\Omega \subset \mathbb{R}^3$ be open and bounded. Let $u_0 : \Omega \to \mathbb{R}$ be a scalar image function.

In Section 3.2.3 we considered segmentations of 2D images and discussed the advantages of region-based active contours over edge-based active contours. The main advantages are: 1) region-based active contours can segment also regions with so-called weak edges, which are edges with only small change in the image intensity function u_0, 2) the gradient of u_0 need not be computed and 3) the method is less sensitive to noise. Therefore, we also restrict on region-based active surfaces for segmentation of 3D images.

We first consider two-phase image segmentation, we consider one closed, orientable surface Γ separating two disjoint regions Ω_1 and Ω_2 such that $\Omega = \Omega_1 \cup \Gamma \cup \Omega_2$. We assume that Γ is oriented by a unit normal vector field $\vec{\nu}$ pointing from Ω_2 to Ω_1.

We consider the analogue of the piecewise constant Mumford-Shah functional (3.20) for an interface *surface* Γ separating now 3D regions Ω_1 and Ω_2. In detail, we search for a surface Γ and for an approximation $u : \Omega \to \mathbb{R}$ of u_0 which is piecewise constant in each region, i.e. $u_{|\Omega_k} = c_k$, $k = 1, 2$, such that

$$E(\Gamma, c_1, c_2) = \sigma|\Gamma| + \lambda \left(\int_{\Omega_1} (u_0 - c_1)^2 \, \mathrm{d}x + \int_{\Omega_2} (u_0 - c_2)^2 \, \mathrm{d}x \right) \tag{5.1}$$

is minimized. This is the Chan-Vese model for 3D images, i.e. an extension of (3.21) with $\mu = 0$ and $\lambda_1 = \lambda_2 = \lambda$ to the three-dimensional case.

As in the two-dimensional case, we first fix the surface Γ and consider variations in the coefficients $c_k \in \mathbb{R}$, $k = 1, 2$. Using the theory of calculus of variations we obtain the mean of the image function in Ω_k for c_k, $k = 1, 2$:

$$c_k = \frac{\int_{\Omega_k} u_0 \, \mathrm{d}x}{\int_{\Omega_k} 1 \, \mathrm{d}x}. \tag{5.2}$$

Then, we fix c_1 and c_2 and consider small variations of the surface Γ by smooth surfaces $\Gamma(t) \subset \Omega$, $t \in (-\epsilon, \epsilon)$, with $\Gamma(0) = \Gamma$. Let $\Omega_1(t)$ and $\Omega_2(t)$ be the regions separated by $\Gamma(t)$. We define

$$f(\vec{x}, c_1, c_2, t) := \begin{cases} (u_0(\vec{x}) - c_1)^2, & \text{if } \vec{x} \in \Omega_1(t), \\ (u_0(\vec{x}) - c_2)^2, & \text{if } \vec{x} \in \Omega_2(t), \end{cases} \tag{5.3}$$

which is defined for a.e. $\vec{x} \in \Omega$.

By using Theorem 2.25 (transport theorem), in particular equations (2.38) and (2.39), we obtain

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}\Big|_{t=0} E(\Gamma(t), c_1, c_2) &= \frac{\mathrm{d}}{\mathrm{d}t}\Big|_{t=0} \left(\sigma \int_{\Gamma(t)} 1 \,\mathrm{d}A + \lambda \int_\Omega f(\vec{x}, c_1, c_2, t) \,\mathrm{d}x \right) \\ &= -\sigma \int_\Gamma \kappa \, V_n \,\mathrm{d}A - \lambda \int_\Gamma \left((u_0 - c_1)^2 - (u_0 - c_2)^2 \right) V_n \,\mathrm{d}A \\ &= -\int_\Gamma (\sigma\kappa + F) \, V_n \,\mathrm{d}A \end{aligned}$$

where $\mathrm{d}A$ is the area element and V_n is the normal velocity (cf. Definition 2.24), κ the curvature (cf. Definition 2.18) and F is an external force given by

$$F(\vec{x}) = \lambda \left((u_0(\vec{x}) - c_1)^2 - (u_0(\vec{x}) - c_2)^2 \right). \tag{5.4}$$

The fastest decrease of the energy is obtained for

$$V_n = \sigma\kappa + F. \tag{5.5}$$

Also multichannel images with a vector-valued image function $\vec{u}_0 : \Omega \to \mathbb{R}^d$ can be handled. This involves vector-valued coefficients $\vec{c}_k$, $k = 1, 2$, and a modification of the external force to, for example,

$$F(\vec{x}) = \sum_{i=1}^{d} \lambda_i \left(((u_0)_i(\vec{x}) - (c_1)_i)^2 - ((u_0)_i(\vec{x}) - (c_2)_i)^2 \right), \tag{5.6}$$

where the subscript i denotes the i-th component of a vector, $i = 1, \ldots, d$. For computation of the coefficients, each component of $\vec{c}_k$ is set to the mean of the corresponding component of $\vec{u}_0$ in the region Ω_k, $k = 1, 2$. In principle, also spaces like the HSV or CB space can be used (cf. Section 3.2.9), where the image function has values in certain submanifolds of $\mathbb{R}^d$. In many practical applications however, for example medical 3D image data generated by computed tomography or magnetic resonance imaging, the image function is often scalar-valued (cf. for example the lung image database of The Cancer Imaging Archive (TCIA), see Reeves et al. (2007); Armato et al. (2011); Reeves and Biancardi (2011)).

5.2.2 Parametric and Multiphase Formulation

Equation (5.5) can be rewritten using a parametric approach to describe the time-dependent surfaces. Further, we now consider a more general setup of multiple surfaces $\Gamma_i(t)$, $t \in [0, T]$, $i = 1, \ldots, N_S$, which separate three-dimensional regions $\Omega_k(t)$, $k = 1, \ldots, N_R$. We assume that the surfaces are compact and oriented by unit normal vector fields $\vec{\nu}_i(\,.\,, t)$ pointing from $\Omega_{k^-(i)}(t)$ to $\Omega_{k^+(i)}(t)$.

Let $\vec{x}_i(\,.\,, t) : \Upsilon_i \to \mathbb{R}^3$, $i = 1, \ldots, N_S$, be a smooth parameterization of $\Gamma_i(t)$, where Υ_i is a two-dimensional reference manifold, for example the sphere $\Upsilon_i = S^2 \subset \mathbb{R}^3$. The normal velocity of $\Gamma_i(t)$ can be expressed as $(V_n)_i = (\vec{x}_i)_t \,.\, \vec{\nu}_i$.

An approximation of the image intensity function u_0 is given by the piecewise constant function $u(\,.\,,t) = \sum_{k=1}^{N_R} c_k(t)\chi_{\Omega_k(t)}$, where $\chi_{\Omega_k(t)}$ is the characteristic function of $\Omega_k(t)$ and $c_k(t)$ is the mean of u_0 in $\Omega_k(t)$.

For each surface, we define the external forcing term

$$F_i(\,.\,,t) = \lambda\left((u_0 - c_{k^+(i)}(t))^2 - (u_0 - c_{k^-(i)}(t))^2\right) \tag{5.7}$$

and obtain the following scheme for the surfaces: Find $\vec{x}_i(\,.\,,t) : \Upsilon_i \to \mathbb{R}^3$ and $\kappa_i(\,.\,,t) : \Upsilon_i \to \mathbb{R}$, $i = 1,\ldots,N_S$, satisfying

$$(\vec{x}_i)_t \,.\, \vec{\nu}_i = \sigma\kappa_i + F_i, \tag{5.8a}$$

$$\Delta_\Gamma \vec{x}_i = \kappa_i \vec{\nu}_i. \tag{5.8b}$$

Equation (5.8a) is a parametric formulation of (5.5) for multiple regions. Equation (5.8b) is identical to (2.21); here it is formulated for the i-th surface. Recall, that Δ_Γ is the Laplace-Beltrami operator, cf. Definition 2.14. Here, we use a small abuse of notation, i.e. we consider κ_i and $\vec{\nu}_i$ as functions defined on Υ_i, i.e. we identify κ_i with $\kappa_i \circ \vec{x}_i$ and $\vec{\nu}_i$ with $\vec{\nu}_i \circ \vec{x}_i$, $i = 1,\ldots,N_S$.

5.2.3 Weak Scheme

We shortly introduce a weak scheme which corresponds to the strong scheme (5.8). It is similar to the two-dimensional case; the integrals over curves are now replaced by integrals over surfaces. We use the notation introduced in Section 5.2.2, i.e. each surface $\Gamma_i(t)$, $i = 1,\ldots,N_S$, is parameterized by an $\vec{x}_i(\,.\,,t) : \Upsilon_i \to \mathbb{R}^3$ with mean curvature $\kappa_i(\,.\,,t) : \Upsilon_i \to \mathbb{R}$.

We define the following weak spaces:

$$W := H^1(\Upsilon_1,\mathbb{R}) \times \ldots \times H^1(\Upsilon_{N_S},\mathbb{R}), \tag{5.9a}$$

$$\underline{V} := H^1(\Upsilon_1,\mathbb{R}^3) \times \ldots \times H^1(\Upsilon_{N_S},\mathbb{R}^3). \tag{5.9b}$$

For $\vec{p} \in \Gamma_i(t)$ let $\vec{g}_{\vec{p}}^{(i)}$ denote the first fundamental form, cf. Definition 2.6, and $(g_{kl}^{(i)})_{k,l}$ the matrix with entries

$$g_{kl}^{(i)}(\vec{q}) := \frac{\partial \vec{x}_i}{\partial q_k}(\vec{q}) \,.\, \frac{\partial \vec{x}_i}{\partial q_l}(\vec{q}), \quad \vec{q} \in \Upsilon_i. \tag{5.10}$$

For $u, v \in L^2(\Upsilon_1,\mathbb{R}^{(3)}) \times \ldots \times L^2(\Upsilon_{N_S},\mathbb{R}^{(3)})$ we define

$$\int_\Gamma u \,.\, v \,\mathrm{d}A := \sum_{i=1}^{N_S} \int_{\Upsilon_i} (u_i \,.\, v_i) \circ \vec{x}_i \sqrt{\det(g_{kl}^{(i)})}\,\mathrm{d}q, \tag{5.11}$$

see also Definition 2.7. For the weak scheme, we consider the following problem: Find time-dependent $\vec{x}(\,.\,,t) \in \underline{V}$, $\kappa(\,.\,,t) \in W$ for $t \in [0,T]$, such that

$$\int_{\Gamma(t)} \vec{x}_t \,.\, \vec{\nu}\,\chi \,\mathrm{d}A - \sigma \int_{\Gamma(t)} \kappa\,\chi \,\mathrm{d}A = \int_{\Gamma(t)} F\,\chi \,\mathrm{d}A, \qquad \forall \chi \in W, \tag{5.12a}$$

$$\int_{\Gamma(t)} \kappa\,\nu \,.\, \vec{\eta} \,\mathrm{d}A + \int_{\Gamma(t)} \nabla_\Gamma \vec{x} \,.\, \nabla_\Gamma \vec{\eta} \,\mathrm{d}A = 0, \qquad \forall \vec{\eta} \in \underline{V}, \tag{5.12b}$$

where $F = (F_1 \circ \vec{x}_1, \ldots, F_{N_S} \circ \vec{x}_{N_S})$.

5.2.4 Related Works

Before describing the numerical framework for solving the scheme (5.8), we shortly mention alternative approaches and related works concerning segmentation of 3D images.

The articles of Cohen (1991) and Cohen and Cohen (1993) belong to the first works, where the active contours model of Kass et al. (1988) is improved (i.a. by a balloon term) and extended to volumetric image data. The authors introduce the analytical framework for a generalization of their balloon model to 3D deformable surfaces. For practical computations however, they suggest to replace a given 3D image by a sequence of 2D images and to apply the 2D active contour model on each single image, followed by a 3D reconstruction of the surface.

An extension of the geodesic active contours model of Caselles et al. (1997a) for 3D image segmentation is proposed by Caselles et al. (1997b). As in the 2D case, the authors use the level set method of Osher and Sethian (1988) to describe the surface implicitly. An example of a segmentation of an image with two linked tori is demonstrated. Further, an example is given, where their method is applied on magnetic resonance images to locate a tumor. The level set method is also used by Yezzi Jr. et al. (1997) who present 2D and 3D active contour models and apply them on medical images. However, practical results are only shown for 2D images.

A detailed literature study on 3D brain cortex segmentation is given in Li et al. (2005). Further, the authors propose a new 3D brain segmentation method based on so-called dual-front active contours which also allow user-interaction. A review on segmentation of medical X-ray computed tomography and magnetic resonance images is given in Sharma and Aggarwal (2010).

Mille (2009) proposes a combination of edge-based and region-based segmentation methods. Both explicit (snakes, triangulated surfaces) and implicit (level set) methods are implemented. For triangulated meshes, a reparameterization for stable node distribution (in 2D and 3D) is performed. Further, for explicit methods a constant global topology is assumed. For images which require topology changes only the level set method is applied.

The Chan-Vese model is used for 2D and 3D medical applications by Rousseau and Bourgault (2009) who perform heart segmentation using an iterative version of the Chan-Vese algorithm. Multiphase segmentation is replaced by iteratively solving a two-phase segmentation problem. Further, the authors employ an additional fidelity term and special initial conditions. The level set method with a finite difference scheme is used to solve the segmentation problem numerically. For 3D applications, they use parallel computation techniques. One main focus of the paper of Rousseau and Bourgault (2009) are the computational details when applying the Chan-Vese model to 2D and 3D medical images.

The level set method is also used by Ardon et al. (2005) and Shen and Huang (2009) for active surfaces. Applications using 3D medical data (i.a. lung and heart segmentation) are considered. Ardon et al. (2005) make use of two so-called constraining curves which are given by the user and should guide the surface evolution. Shen and Huang (2009) employ surface-distance-based functions in order to use region information combined with spatial constraints. Mikula et al. (2011) use the level-set method for 3D cell membrane segmentation. They use a level set equation for advective motion to detect the center of cells and a generalized subjective surface model (cf. Sarti et al. (2002)) to detect the inner cell boundaries. An approach for tracking cells in 4D images (3D data + time) has been recently developed by Mikula et al. (2014a).

Most of the works on 3D image segmentation with active contours make use of the level set method. In contrast to them, we present a parametric approach for 3D image segmentation. Our method will also allow for topology changes by efficiently detecting them and performing modifications of the surface triangulation.

From a computational and numerical point of view, different approaches for parametric surface evolution exist in the literature:

Brakke (1992) presents a program called "The Surface Evolver" which computes evolving triangulated surfaces, where the evolution is driven by energy minimization problems with possible constraints. The program is able to perform topology changes like splitting if this is instructed by the user.

Brochu and Bridson (2009) propose another explicit method for evolving surfaces by using triangulated surfaces. The authors propose several techniques for mesh quality improvement like edge splitting or flipping and techniques for topology changes. Splitting is done by a combination of removing degenerate elements and a certain mesh separation method based on duplication and separation of nodes. Surfaces are thus splitted if they become locally too thin. Merging is detected by searching for edges which are too close. Each triangle at those edges is deleted resulting in two surfaces with each a temporary hole. These holes are closed by creation of new triangles. In our work we make use of the idea of deleting incident triangles and creating new triangles at the intermediate holes. The detection in our case will be an extension of the method of Mikula and Urbán (2012) and Balažovjech et al. (2012) (recall also Section 3.3.5) from 2D planar curves to topology changes of surfaces in $\mathbb{R}^3$.

Finite element approaches for surface evolutions are pursued by Bänsch et al. (2005). They do not consider image segmentation applications or mean curvature flow, but surface diffusion. They rewrite the surface diffusion problem to a scheme for the mean curvature, the mean curvature vector, the velocity and the velocity vector. Several mesh quality routines like mesh regularization (keeping all angles of simplices at a node of the same size), time step control, refine/coarsening routines and angle width control are proposed.

The finite element scheme in this paper (cf. Section 5.3) is based on the work of Barrett et al. (2008b) where the authors consider surface diffusion, (inverse) mean curvature flow and non-linear flows. They propose a parametric finite element approximation and perform various numerical experiments including surface diffusion with a pinch-off. The computation is stopped when a pinch-off occurs. In Barrett et al. (2008c), the authors consider Willmore flow and related flows and present a parametric finite element approximation. Further, a method is developed for mesh regularization. The idea is to reduce or induce the tangential motion of nodes. We will use this method for 3D image segmentation to avoid problems concerning the mesh quality. Especially close to topology changes like splitting, we will observe that some control over the tangential motion of the nodes is necessary.

5.3 Numerical Approximation

5.3.1 Finite Element Approximation

We introduce a finite element approximation for the scheme (5.8).

Let $0 = t_0 < t_1 < \ldots < t_M = T$ be a decomposition of the time interval into possibly variable time steps $\tau_m = t_{m+1} - t_m$ for $m = 0, \ldots, M-1$.

Let N_S denote the number of surfaces and N_R denote the number of regions. Let the smooth surface $\Gamma_i(t_m)$, $i = 1, \ldots, N_S$, be approximated by a polyhedral surface Γ_i^m of the form

$$\Gamma_i^m = \bigcup_{j=1}^{N_{i,F}} \overline{\sigma_{i,j}^m}, \tag{5.13}$$

where $\sigma_{i,j}^m$, $j = 1, \ldots, N_{i,F}$, are disjoint, open simplices (also called faces) with vertices $\vec{q}_{i,j}^m$, $j = 1, \ldots, N_{i,V}$. Further, let $h := \max_{i=1,\ldots,N_S, j=1,\ldots,N_{i,F}} \mathrm{diam}(\sigma_{i,j}^m)$ be the maximum diameter of a simplex of the triangulated surfaces. The diameter $\mathrm{diam}(\sigma_{i,j}^m)$ is defined as the maximum distance between two points of $\overline{\sigma_{i,j}^m}$.

Let $N_{i,E}$ be the number of edges of the triangulation and g_i the genus of Γ_i^m. The Euler characteristic

$$\chi_i = N_{i,V} - N_{i,E} + N_{i,F} = 2 - 2g_i \tag{5.14}$$

relates the number of vertices, edges and faces with the genus of a surface.

Let $\vec{X}_i^m$ be a parameterization of Γ_i^m and let Ω_k^m, $k = 1, \ldots, N_R$, denote the open, disjoint subsets of Ω separated by Γ_i^m, $i = 1, \ldots, N_S$. Thus, Ω_k^m is an approximation of $\Omega_k(t_m)$ for $k = 1, \ldots, N_R$.

The new surfaces Γ_i^{m+1} are parameterized over Γ_i^m. Therefore, we define the following finite element space

$$\begin{aligned} W(\Gamma^m) := \Big\{ &(\eta_1, \ldots, \eta_{N_S}) \in C(\Gamma_1^m, \mathbb{R}) \times \ldots \times C(\Gamma_{N_S}^m, \mathbb{R}) \,:\, \eta_i|_{\sigma_{i,j}^m} \text{ is linear}, \\ &\forall i = 1, \ldots, N_S,\, j = 1, \ldots, N_{i,F} \Big\}, \end{aligned} \tag{5.15a}$$

$$\begin{aligned} \underline{V}(\Gamma^m) := \Big\{ &(\vec{\eta}_1, \ldots, \vec{\eta}_{N_S}) \in C(\Gamma_1^m, \mathbb{R}^3) \times \ldots \times C(\Gamma_{N_S}^m, \mathbb{R}^3) \,:\, \vec{\eta}_i|_{\sigma_{i,j}^m} \text{ is linear}, \\ &\forall i = 1, \ldots, N_S,\, j = 1, \ldots, N_{i,F} \Big\}. \end{aligned} \tag{5.15b}$$

The spaces $W(\Gamma^m)$ and $\underline{V}(\Gamma^m)$ thus consist of scalar or vector-valued, piecewise linear functions defined on Γ^m.

A basis of $W(\Gamma^m)$ is given by functions $\chi_{i,j}^m := ((\chi_{i,j}^m)_1, \ldots, (\chi_{i,j}^m)_{N_S}) \in W(\Gamma^m)$, where $(\chi_{i,j}^m)_k(\vec{q}_{k,l}^m) = \delta_{ik}\delta_{jl}$ for $i, k = 1, \ldots, N_S$, $j = 1, \ldots, N_{i,V}$, $l = 1, \ldots, N_{k,V}$.

Note, that for $m \geq 1$, $\vec{X}^m \in \underline{V}(\Gamma^{m-1})$ and for $m \geq 0$, $\vec{X}^m \in \underline{V}(\Gamma^m)$ is the identity defined on Γ^m.

For scalar and vector-valued functions $u = (u_1, \ldots, u_{N_S})$, $v = (v_1, \ldots, v_{N_S}) \in L^2(\Gamma_1^m, \mathbb{R}^{(3)}) \times \ldots \times L^2(\Gamma_{N_S}^m, \mathbb{R}^{(3)})$, we introduce the L^2-inner product over the current polyhedral surface Γ^m as follows:

$$\langle u, v \rangle_m := \int_{\Gamma^m} u \,.\, v \,\mathrm{d}A = \sum_{i=1}^{N_S} \int_{\Gamma_i^m} u_i \,.\, v_i \,\mathrm{d}A. \tag{5.16}$$

If u, v are piecewise continuous with possible jumps across the edges of $\sigma_{i,j}^m$, $i = 1, \ldots, N_S$, $j = 1, \ldots, N_{i,F}$, the mass lumped inner product is defined as

$$\langle u, v \rangle_m^h := \frac{1}{3} \sum_{i=1}^{N_S} \sum_{j=1}^{N_{i,F}} |\sigma_{i,j}^m| \sum_{l=1}^{3} (u \,.\, v)((\vec{q}_{i,j_l}^m)^-), \tag{5.17}$$

where $\vec{q}_{i,j_l}^m$, $l = 1, 2, 3$, are the vertices of $\sigma_{i,j}^m$, $|\sigma_{i,j}^m| = \frac{1}{2}\|(\vec{q}_{i,j_2}^m - \vec{q}_{i,j_1}^m) \times (\vec{q}_{i,j_3}^m - \vec{q}_{i,j_1}^m)\|$ is the area of $\sigma_{i,j}^m$ and $u((\vec{q}_{i,j_l}^m)^-) := \lim_{\vec{p} \to \vec{q}_{i,j_l}^m, \vec{p} \in \sigma_{i,j}^m} u(\vec{p})$.

We assume that the vertices $\left\{\vec{q}^{\,m}_{i,j_l}\right\}_{l=1}^{3}$, $j = 1, \dots N_{i,F}$, are ordered such that the outer unit normal $\vec{\nu}^{\,m}_i$ at Γ^m_i is given by

$$\vec{\nu}^{\,m}_i|_{\sigma^m_{i,j}} := \vec{\nu}^{\,m}_{i,j} := \frac{(\vec{q}^{\,m}_{i,j_2} - \vec{q}^{\,m}_{i,j_1}) \times (\vec{q}^{\,m}_{i,j_3} - \vec{q}^{\,m}_{i,j_1})}{\|(\vec{q}^{\,m}_{i,j_2} - \vec{q}^{\,m}_{i,j_1}) \times (\vec{q}^{\,m}_{i,j_3} - \vec{q}^{\,m}_{i,j_1})\|}. \tag{5.18}$$

We propose the following finite element scheme approximating the scheme (5.8): Let Γ^0 be a union of polyhedral surfaces approximating $\Gamma(0)$ and let $\vec{X}^0 \in \underline{V}(\Gamma^0)$ be the identity function on Γ^0. Find $\vec{X}^{m+1} \in \underline{V}(\Gamma^m)$ and $\kappa^{m+1} \in W(\Gamma^m)$, $m = 0, 1, \dots, M-1$, such that

$$\langle \frac{\vec{X}^{m+1} - \vec{X}^m}{\tau_m}, \chi\, \vec{\nu}^{\,m} \rangle^h_m - \sigma \langle \kappa^{m+1}, \chi \rangle^h_m = \langle F^m, \chi \rangle^h_m, \qquad \forall \chi \in W(\Gamma^m), \tag{5.19a}$$

$$\langle \kappa^{m+1}\, \vec{\nu}^{\,m}, \vec{\eta} \rangle^h_m + \langle \nabla_s \vec{X}^{m+1}, \nabla_s \vec{\eta} \rangle_m = 0, \qquad \forall \vec{\eta} \in \underline{V}(\Gamma^m). \tag{5.19b}$$

Here, $F^m = (F^m_1, \dots, F^m_{N_S})$ is defined by

$$F^m_i(\vec{q}^{\,m}_{i,j}) := \lambda \left((u_0(\vec{X}^m_i(\vec{q}^{\,m}_{i,j})) - c^m_{k^+(i)})^2 - (u_0(\vec{X}^m_i(\vec{q}^{\,m}_{i,j})) - c^m_{k^-(i)})^2 \right),$$

for $i = 1, \dots, N_S$ and $j = 1, \dots, N_{i,V}$ and c^m_k is set to the mean of u_0 in Ω^m_k for $k = 1, \dots, N_R$.

We further introduce a weighted normal defined at the nodes $\vec{X}^m_i(\vec{q}^{\,m}_{i,j}) = \vec{q}^{\,m}_{i,j} \in \Gamma^m_i$ by setting

$$\vec{\omega}^m_i(\vec{q}^{\,m}_{i,j}) := \vec{\omega}^m_{i,j} := \frac{1}{|\Lambda^m_{i,j}|} \sum_{\sigma^m_{i,l} \in \mathcal{T}^m_{i,j}} |\sigma^m_{i,l}|\, \vec{\nu}^{\,m}_{i,l}, \tag{5.20}$$

where for $i = 1, \dots, N_S$ and $j = 1, \dots, N_{i,V}$, $\mathcal{T}^m_{i,j} := \left\{ \sigma^m_{i,l} : \vec{q}^{\,m}_{i,j} \in \overline{\sigma^m_{i,l}} \right\}$ and $\Lambda^m_{i,j} := \bigcup_{\sigma^m_{i,l} \in \mathcal{T}^m_{i,j}} \overline{\sigma^m_{i,l}}$.

Further, we set $\vec{v}^m_i(\vec{q}^{\,m}_{i,j}) := \vec{v}^m_{i,j} := \vec{\omega}^m_{i,j}/\|\vec{\omega}^m_{i,j}\|$ and $\vec{\omega}^m = (\vec{\omega}^m_1, \dots, \vec{\omega}^m_{N_S})$ and $\vec{v}^m = (\vec{v}^m_1, \dots, \vec{v}^m_{N_S})$.

As in Barrett, Garcke, and Nürnberg (2008a,b), we make a very mild assumption on the triangulations:

($\mathcal{A}$) For $m = 0, \dots, M$, we assume that $|\sigma^m_{i,j}| > 0$ for all $i = 1, \dots, N_S$ and $j = 1, \dots, N_{i,F}$ and for $m = 0, \dots, M-1$, we assume that $\dim \operatorname{span} \left\{ \vec{\omega}^m_{i,j} \right\}_{j=1}^{N_{i,V}} = 3$.

The assumption ($\mathcal{A}$) is only violated in very rare cases. For closed surfaces without self intersections, it always holds.

Theorem 5.1. *Let the assumption ($\mathcal{A}$) hold. Then there exists a unique solution* $\left\{ \vec{X}^{m+1}, \kappa^{m+1} \right\} \in \underline{V}(\Gamma^m) \times W(\Gamma^m)$ *to the system* (5.19).

Proof. (Cf. Barrett, Garcke, and Nürnberg (2008b)) Since the system is linear, it is sufficient to show uniqueness. Therefore, we consider the following scheme: Find $\vec{X} \in \underline{V}(\Gamma^m)$ and $\kappa \in W(\Gamma^m)$ such that

$$-\frac{1}{\tau_m} \langle \vec{X}, \chi\, \vec{\nu}^{\,m} \rangle^h_m + \sigma \langle \kappa, \chi \rangle^h_m = 0, \qquad \forall \chi \in W(\Gamma^m), \tag{5.21a}$$

$$\langle \kappa\, \vec{\nu}^{\,m}, \vec{\eta} \rangle^h_m + \langle \nabla_s \vec{X}, \nabla_s \vec{\eta} \rangle_m = 0, \qquad \forall \vec{\eta} \in \underline{V}(\Gamma^m), \tag{5.21b}$$

holds. Testing (5.21a) with $\chi = \kappa$ and (5.21b) with $\vec{\eta} = \vec{X}$ leads to

$$\sigma\tau_m \langle \kappa, \kappa \rangle_m^h + \langle \nabla_s \vec{X}, \nabla_s \vec{X} \rangle_m = 0. \tag{5.22}$$

It follows that $\kappa_{i,j} = 0$ and $\vec{X}_{i,j} = \vec{C}_i \in \mathbb{R}^3$ for $i = 1, \dots, N_S$, $j = 1, \dots, N_{i,V}$. Inserting $\kappa = 0$ and $\vec{X} = \vec{C} = (\vec{C}_1, \dots, \vec{C}_{N_S})$ in (5.21a) results in

$$\langle \vec{C}, \chi\, \vec{\nu}^m \rangle_m^h = 0, \quad \forall \chi \in W(\Gamma^m). \tag{5.23}$$

Choosing $\chi = \chi_{i,j}^m$ (the standard basis) and using (5.20), the definition of $\vec{\omega}_{i,j}^m$, results in

$$\vec{C}_i \,.\, \vec{\omega}_{i,j}^m = 0, \quad \forall i = 1, \dots, N_S,\, j = 1, \dots, N_{i,V}. \tag{5.24}$$

Finally, using the assumption $(\mathcal{A})$, it follows that $\vec{C}_i = 0$ for each $i = 1, \dots, N_S$. □

5.3.2 Solution of the Discrete System

We define $\delta\vec{X}^{m+1} := \vec{X}^{m+1} - \vec{X}^m$. As $\delta\vec{X}^{m+1}$ and κ^{m+1} are uniquely given by their values at the nodes $\vec{q}_{i,j}^m$, we consider them as elements in $(\mathbb{R}^3)^N$ and $\mathbb{R}^N$, respectively, where $N = \sum_{i=1}^{N_S} N_{i,V}$. We introduce the matrices $M_m \in \mathbb{R}^{N\times N}$, $\vec{N}_m \in (\mathbb{R}^3)^{N\times N}$ and $\vec{A}_m \in (\mathbb{R}^{3\times 3})^{N\times N}$ by

$$M_m := \begin{pmatrix} M_m^1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & M_m^{N_S} \end{pmatrix}, \vec{N}_m := \begin{pmatrix} \vec{N}_m^1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \vec{N}_m^{N_S} \end{pmatrix}, \vec{A}_m := \begin{pmatrix} \vec{A}_m^1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \vec{A}_m^{N_S} \end{pmatrix},$$

where $M_m^i \in \mathbb{R}^{N_{i,V}\times N_{i,V}}$, $\vec{N}_m^i \in (\mathbb{R}^3)^{N_{i,V}\times N_{i,V}}$, $\vec{A}_m^i \in (\mathbb{R}^{3\times 3})^{N_{i,V}\times N_{i,V}}$, $i = 1, \dots, N_S$. Their entries are defined by

$$[M_m^i]_{kl} := \langle \chi_{i,k}^m, \chi_{i,l}^m \rangle_m^h, \quad [\vec{N}_m^i]_{kl} := \langle \chi_{i,k}^m, \chi_{i,l}^m \vec{\nu}^m \rangle_m^h, \quad [\vec{A}_m^i]_{kl} = \langle \nabla_s \chi_{i,k}^m, \nabla_s \chi_{i,l}^m \rangle_m \vec{\mathrm{Id}}_3, \tag{5.25}$$

with $i = 1, \dots, N_S$, $k, l = 1, \dots, N_{i,V}$. Here, $\vec{\mathrm{Id}}_3$ denotes the identity matrix in $\mathbb{R}^{3\times 3}$. Further, we introduce $b_m = (b_m^1, \dots, b_m^{N_S}) \in \mathbb{R}^N$ defined by

$$[b_m^i]_k := \langle F_i^m, \chi_{i,k}^m \rangle_m^h, \quad i = 1, \dots, N_S,\, k = 1, \dots, N_{i,V}. \tag{5.26}$$

The scheme (5.19) can be rewritten to the following problem: Let Γ^0 be a polyhedral approximation of $\Gamma(0)$ and let $\vec{X}^0 = (\vec{X}_1^0, \dots, \vec{X}_{N_S}^0) \in (\mathbb{R}^3)^N$ with $\vec{X}_i^0 = (\vec{X}_{i,1}^0, \dots, \vec{X}_{i,N_{i,V}}^0)$ such that $\vec{X}_{i,j}^0$ are the coordinates of the vertices of Γ_i^0 for $i = 1, \dots, N_S$, $j = 1, \dots, N_{i,V}$. For $m = 0, \dots, M-1$ find $\delta\vec{X}^{m+1} \in (\mathbb{R}^3)^N$ and $\kappa^{m+1} \in \mathbb{R}^N$ such that

$$\begin{pmatrix} \sigma\tau_m M_m & -\vec{N}_m^T \\ \vec{N}_m & \vec{A}_m \end{pmatrix} \begin{pmatrix} \kappa^{m+1} \\ \delta\vec{X}^{m+1} \end{pmatrix} = \begin{pmatrix} -\tau_m b_m \\ -\vec{A}_m \vec{X}^m \end{pmatrix}. \tag{5.27}$$

Applying a Schur complement approach, we can transform this system to

$$\kappa^{m+1} = \frac{1}{\sigma} M_m^{-1} \left(\frac{1}{\tau_m} \vec{N}_m^T \delta\vec{X}^{m+1} - b_m \right), \tag{5.28a}$$

$$\left(\frac{1}{\sigma\tau_m} \vec{N}_m M_m^{-1} \vec{N}_m^T + \vec{A}_m \right) \delta\vec{X}^{m+1} = -\vec{A}_m \vec{X}^m + \frac{1}{\sigma} \vec{N}_m M_m^{-1} b_m. \tag{5.28b}$$

Since the system matrix in (5.28b) is symmetric and positive definite under the assumption $(\mathcal{A})$, there exists a unique solution.

5.3.3 Topology Changes

Parametric methods cannot handle topology changes automatically in contrast to other numerical methods like the level set method. During the evolution of surfaces singularities can occur like a pinch-off, see Bänsch, Morin, and Nochetto (2005); Barrett, Garcke, and Nürnberg (2008b). In order to proceed after a pinch-off, the surface has to be split in two single surfaces. Other possible topology changes, that we will consider here, are merging of two surfaces and change of the genus (examples: torus to sphere, sphere to torus).

Mikula and Urbán (2012) propose an algorithm to efficiently detect splitting and merging of evolving curves in $\mathbb{R}^2$, see also Balažovjech et al. (2012). Our algorithm to detect topology changes in 2D (cf. Section 3.3.5) was based on this work. We want to generalize this approach to the 3D case, i.e. we want to detect topology changes which could occur during the evolution of surfaces. Having found the location where a topology change occurs, we propose a method how to modify the triangulations.

Detection of a Topology Change

The idea is to construct a uniform 3D grid of cubes. A topology change may occur, if a large number of nodes or if nodes of different surfaces or different parts of one surface (with opposite normal vector) are located in one cube.

In detail, to detect a change in topology, we construct a uniform 3D background grid which covers the image domain Ω. In the following, we assume that Ω is a cuboid. If the 3D image u_0 is not given on a cuboid volume, we consider a cuboid which contains Ω.

Let $\Omega = [x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}] \times [z_{\min}, z_{\max}] \subset \mathbb{R}^3$ be the image domain containing in particular the surfaces Γ_i^m, $i = 1, \dots, N_S$. We consider a grid dividing Ω in a set of many small cubes of edge width $a \in \mathbb{R}$. Let the grid consist of $N_x \times N_y \times N_z$ cubes, where $N_x = \text{ceil}((x_{\max} - x_{\min})/a)$, $N_y = \text{ceil}((y_{\max} - y_{\min})/a)$ and $N_z = \text{ceil}((z_{\max} - z_{\min})/a)$.

We now perform one loop over all surfaces and nodes $\vec{X}_{i,j}^m$, $i = 1, \dots, N_S$, $j = 1, \dots, N_{i,V}$. If a node $\vec{X}_{i,j}^m$ is the first node which is detected to lie in a certain cube, we create a new list for that cube, where we store the index pair (i, j). If another node has already been identified to be located inside that cube, we add the index pair (i, j) to the existing list.

If there is a large number of nodes located in a cube, i.e. more than N_{detect} nodes, a topology change likely occurs, and the cube is stored in a list for possible topology changes. It is also possible that the node density is only locally very high at this location, but no topology change happens. Note, that in contrast to the two-dimensional case and the case of curves on surfaces, we cannot prove an equidistribution property. Thus, we cannot exclude such cases in which nodes are locally very dense and located in one cube by purely adapting the grid size a as in the case of curves. But we can use the surface index and the weighted normal vectors $\vec{\omega}_{i,j}^m$ to detect probable topology changes, see below.

If there are less than N_{detect} nodes in the current list, we compare the surface index and the direction of the weighted normal vector of the current node with those of the nodes already stored in the list. If two different surface indices i_1 and i_2 occur or if two nodes with (nearly) opposite weighted normal vector are located in one cube, a topology change likely happens. The corresponding cube is accordingly stored in a list for topology changes.

After having considered all nodes, the cubes marked for topology changes are considered one by one. If a topology change is identified, the surface triangulation is accordingly changed.

Thus, by successively considering all marked cubes, more than one topology change can be executed in one time step. The list can be optionally sorted such that the cube with the largest number of nodes is considered first.

Identification of the Topology Change

For identifying which kind of topology change occurs the nodes of the affected cube, i.e. the current cube of the sorted list, and the nodes of 26 neighbor cubes (in total 27 cubes, i.e. $3 \times 3 \times 3$ cubes) are considered. Let $S = \{j_1, \ldots, j_{n_c}\}$ denote the index set of the nodes and let $\vec{X}_j$, $j \in S$, denote the coordinates of the nodes located in the cubes. Further, let $\vec{\omega}_j$, $j \in S$, denote the corresponding weighted normal vectors. For the ease of illustration, we omit the time dependency (time index m) in the notation.

The different topology changes are distinguished by considering the weighted normal vectors $\vec{\omega}_j$. The idea is that in case of merging and increasing genus, their are two main group of nodes which can be found by considering their normal vectors. For splitting and decrease of genus, there are more than two main directions.

The node with index $j = j_1$ is set as representative of the first group. We choose thresholds $thr1 < thr2$, for example $thr1 = 20°$, $thr2 = 160°$. For $j = j_2, \ldots, j_{n_c}$ we consider the angle α between $\vec{\omega}_{j_1}$ and $\vec{\omega}_j$. If $\alpha < thr1$, the node j belongs to the first group. If $\alpha > thr2$ and if the second group is empty, the node j becomes the representative node of the second group. If the second group is not empty, we consider the angle β between $\vec{\omega}_j$ and $\vec{\omega}_{k_0}$, where $k_0 \in \{j_2, \ldots, j_{n_c}\}$ is the representative of the second group. If $\beta < thr1$, j is added to the second group.

For the next search we replace $\vec{\omega}_{j_1}$ and $\vec{\omega}_{k_0}$ by the average normal vectors $\vec{n}_1$ and $\vec{n}_2$ of group 1 and group 2, respectively, and re-consider the nodes which could not have been assigned to one group in the first step. If the angle between $\vec{\omega}_j$ and $\vec{n}_1$ or $\vec{\omega}_j$ and $\vec{n}_2$ is smaller than $thr1$, the node $\vec{X}_j$ is added to the corresponding group.

If group 2 is empty, or if one of the groups consists of only a small number of nodes (e.g. $< 5\%$ of n_c), we start again by using another node as representative for the first group (e.g. $j = j_2$), since the node $j = j_1$ could be an outlier. If no start node can be found, such that there exist two groups of nodes as described above, no topology change takes place. This can happen, if all weighted normals point in nearly the same direction.

The method such provides that a topology change like splitting is not wrongly detected at locations where several nodes are just close to each other but their normal vectors point in the same direction. From a mesh quality point of view, such meshes should be avoided; the nodes should be distributed equally over the surface. However, the detection of topology changes should be robust enough not to wrongly detect a splitting at locations where there is just a high density of nodes.

If both groups have a sufficient number of nodes, we proceed by considering the remaining normal vectors which could have not been assigned to one of the two groups. We again consider the angle between $\vec{\omega}_j$ and $\vec{n}_1$ and $\vec{\omega}_j$ and $\vec{n}_2$. If both angles are $> thr3$ (e.g. $thr3 = 40°$), the normal $\vec{\omega}_j$ points in a complete different direction compared to $\vec{n}_1$ and $\vec{n}_2$. Let N_0 be the number of such points. If N_0 exceed a predefined number (e.g. $1/3\, n_c$), then there are more than two main groups of directions. In this case, there is a splitting or decrease of genus. Splitting and decrease of genus are handled similarly (see below for details). Triangles close to the detected cube are deleted. If the remaining triangulation of

the former surface consists of two connected components, a splitting occurs. If the remaining triangulation is connected, a decrease of genus occurs.

If N_0 is zero or if it does not exceed the predefined number, possible remaining normal vectors are only single outliers and there are only two main groups of normal vectors with nearly opposed normal vector. In this case, there is a merging or increase of genus. If there are nodes belonging to two different surfaces, a merging occurs. Otherwise, an increase of genus occurs.

A summary and an illustration of the algorithm to detect and identify topology changes is given in Figure 5.1.

Algorithm for Splitting and Decreasing of Genus

We propose the following algorithm for a possible modification of the surface triangulation after the detection of a splitting or decrease of genus.

- **Preparation and deletion of simplices:** In case of splitting or decreasing genus, we consider the set of affected nodes $\vec{X}_j$, $j \in S$, which are located in the cube or in a neighbor cube, where the topology change has been detected. Let $\vec{p}_E$ be the mean of the points $\{\vec{X}_j : j \in S\}$. We delete all simplices with at least one vertex belonging to the set $\{\vec{X}_j : j \in S\}$. When deleting one simplex, we change the neighbor information of neighbor simplices at the corresponding edges to -1 (free edges). Simplices with two or three free edges are deleted as well, see Figure 5.2. Deletion creates two temporary holes in the surface(s). As a result we either have two sets of connected simplices ($\rightarrow$ splitting) or one set of connected simplices ($\rightarrow$ decrease of genus).

- **Set surface index (splitting only):** The remaining simplices with one free edge form two connected sets. For one set, we need to re-set the surface index. Let i be the surface index of the original surface. By splitting the total number of surfaces is increased to $N_S + 1$. Starting with one simplex with a free edge, we re-set its surface index from i to $N_S + 1$. Then, we consider its neighbor simplices and assign them to surface $N_S + 1$ also. This procedure is similar to the assignment of region indices in case of images on surfaces, recall Section 4.3.5 and Figure 4.3: The simplices of one of the two connected components are assigned one by one to the surface $N_S + 1$ by heritage, i.e. by use of neighbor information.

- **Generate new simplices:** We close each of the two intermediate holes (where simplices have been deleted) by constructing new simplices at edges of simplices where the neighbor information has been set to -1. First we create two new points, each with coordinates $\vec{p}_E$, one for each intermediate hole. In the next time steps the two nodes can move away from each other. If σ is a simplex with a free edge given by $\vec{X}_{\sigma,j_1}$ and $\vec{X}_{\sigma,j_2}$ with no neighbor simplex, a new simplex is generated given by the vertices $\vec{X}_{\sigma,j_1}$, $\vec{X}_{\sigma,j_2}$ and one of the two new nodes at $\vec{p}_E$. The new simplex inherits the surface index from σ.

- **Improve mesh quality:** By simply connecting all free edges with one of the two new vertices at $\vec{p}_E$, the two vertices can belong to a big number of simplices. Let σ_1 and σ_2 be two of the newly generated simplices with one common edge. We construct 4 new simplices from σ_1 and σ_2 such that the new vertex belongs to only one of the 4 new simplices, see Figure 5.3. Therefore the number of elements to which the vertices at $\vec{p}_E$

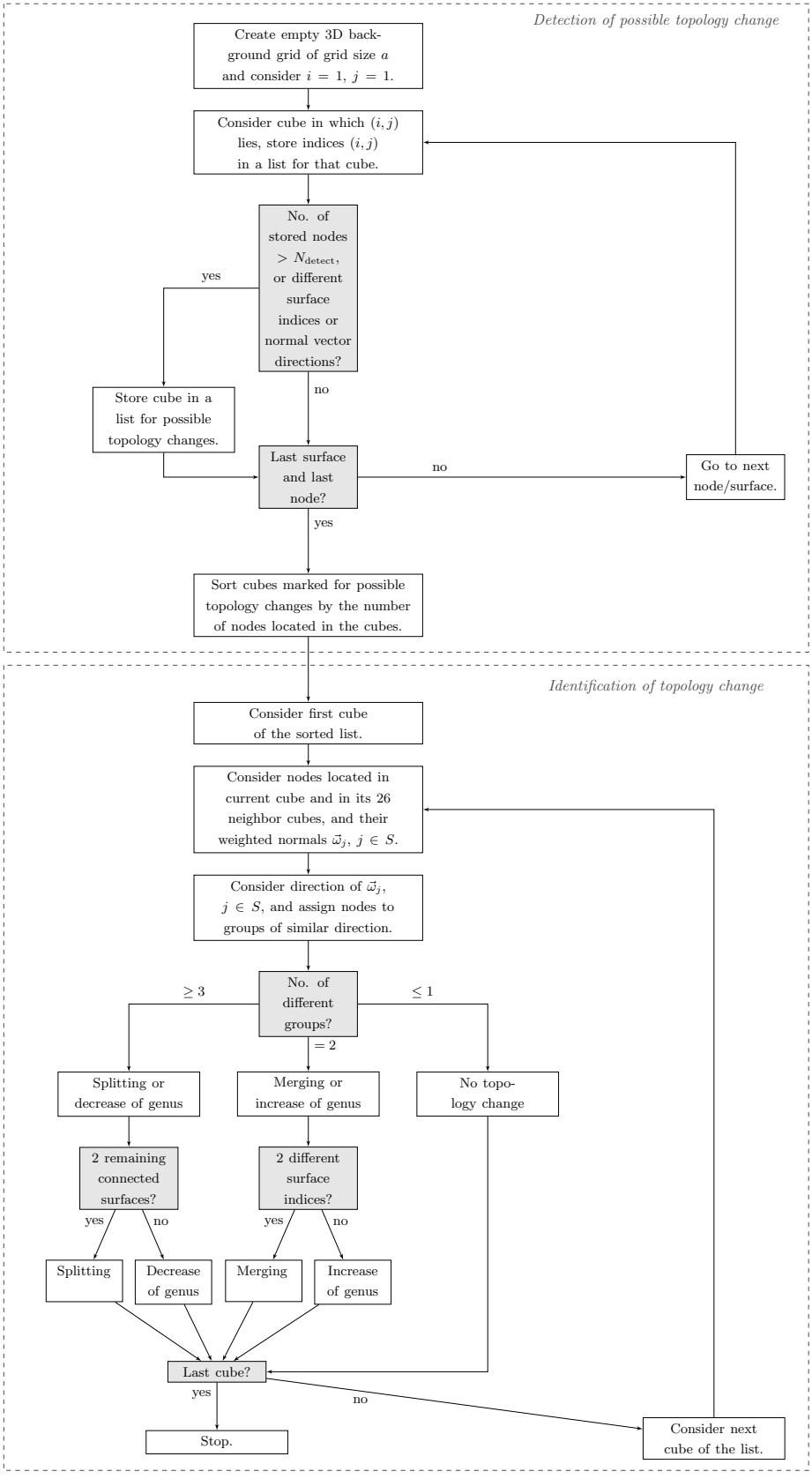


Figure 5.1: Illustration of the detection and identification of topology changes of surfaces.

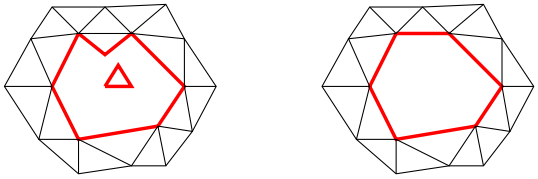


Figure 5.2: Left: Part of the surface near a temporary hole. Free edges are drawn with red color. Right: Surface near the hole after deletion of simplices with more than one free edge.

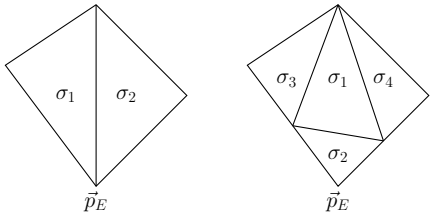


Figure 5.3: Improving mesh quality after a splitting or decrease of genus - Decrease the number of edges at the node $\vec{p}_E$.

belong to is approximately halved. We repeat this procedure until each of the newly created nodes at $\vec{p}_E$ belongs to ≤ 8 elements.

Algorithm for Merging and Increasing of Genus

We propose the following algorithm for a possible modification of the surface triangulation after the detection of a merging or increase of genus.

- **Preparation and deletion of simplices:** In case of merging or increasing genus, we consider the set of affected nodes $\vec{X}_j$, $j \in S$, which are located in the cube or in a neighbor cube, where the topology change has been detected. We delete all simplices with at least one vertex belonging to the set $\{\vec{X}_j : j \in S\}$. This generates temporary free edges, i.e. simplices exist which do not have a neighbor simplex at that edge. The neighbor index corresponding to the free edge (formally the index of a simplex which has been deleted) is set to -1. This creates two intermediate holes. Simplices with two or three free edges are deleted as well, see Figure 5.2.

- **Matching free nodes/edges:** There exist now two sets of connected free edges. Let $I_{free,k}$, $k = 1, 2$, be the set of nodes corresponding to the free edges (end points of the edges). We try to match the nodes of $I_{free,1}$ with the nodes of $I_{free,2}$ using the Hungarian method of Kuhn (1955) which is a combinatorial optimization algorithm. The Euclidean distance is used as cost criterion for matching two nodes. Since the number of nodes of the two sets need not be equal, there can be nodes which could not have been matched in the first step, see Figure 5.4 (left). Therefore, new nodes are created by bisection of simplices at a free edge. Finally, each node can be matched, see Figure 5.4 (right).

- **Point/Edge merging:** Since two matched nodes can have slightly different coordinates, they are replaced by one node in the middle of the line connecting the two nodes.

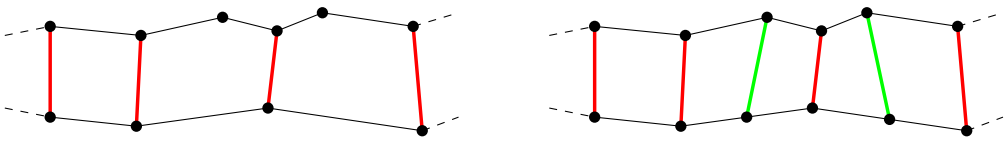


Figure 5.4: A subset of the free nodes. Left: Intermediate matching (red lines mark matching pairs). Right: Complete matching after inserting new nodes (red and green lines mark matching pairs).

The free edges of the two open holes are merged by identifying the matched nodes. The simplices, nodes and edges administration needs to be adapted. The nodes in $I_{free,1}$ are updated; each point is replaced by the mid point between it and its matching partner. The nodes belonging to $I_{free,2}$ are deleted. Half of the free edges are deleted. The edge, node and neighbor information of the simplices at the former holes need to be adapted. In case of merging, the surface index of all simplices belonging to the second surface is set to the surface index of the first surface.

Note, that local refinement after a merging is typically necessary. This is automatically done, by a refinement method described in Section 5.3.4. If the surface grows locally near the former merging part, the simplices will become greater compared to the average simplex of the surface. In this case, the large simplices will be refined.

The idea of creating two intermediate open holes and merging the two surfaces there is based on the work of Brochu and Bridson (2009). In their method, however, each hole is restricted to consist of exactly four free edges. New triangles between the free edges are created instead of merging the edges.

In our method, the seeking for close points (and therefore close edges/simplices) is very efficient, since we make use of a background grid motivated by the method of Mikula and Urbán (2012). We extended their method originally intended for curves in the plane to topology changes of surfaces. We allow for intermediate holes with an arbitrary number of free edges. The hole size is of the magnitude of the grid size.

In the work of Brochu and Bridson (2009) a variety of additional topological operations are proposed like edge operations (edge split, edge flip) and operations to avoid simplex collision.

The operations we have implemented beyond the detection of topology changes are described in the next sections. Mesh operations are only needed in special situations, for example shortly before or after topology changes. In many situations, our method automatically provides a good mesh quality.

5.3.4 Additional Computational Aspects

Computations of regions and coefficients

The computation of regions Ω_k^m and coefficients c_k^m is done similarly as in the two-dimensional case, recall Section 3.3.6.

We assign each voxel of the three-dimensional image domain to a phase Ω_k^m. If a voxel is truncated by a surface, it is assigned to the phase to which the largest part belongs or to

any of the two regions in case of two equal parts. Let S_k^m be the set of n_k^m voxels belonging to Ω_k^m. Then the approximation c_k^m is set to

$$c_k^m := \frac{C_k^m}{n_k^m}, \qquad C_k^m := \sum_{vox \in S_k^m} u_0|_{vox}. \tag{5.29}$$

As in the two-dimensional case, the entire image domain needs to be considered only for $m = 0$. For $m > 0$, we only locally update the regions and re-compute the coefficients on this basis. In the two-dimensional case, we considered a small band of pixels around the current curves. For the three-dimensional case, we can generalize this concept to 3D regions by considering a small tube of voxels around the current surfaces.

As the normal $\vec{\nu}_i^m$ points from $\Omega_{k^-(i)}^m$ to $\Omega_{k^+(i)}^m$, the voxels close to the surface Γ_i^m can be assigned to the phase $k^+(i)$ or $k^-(i)$, respectively.

In the update step, we first set $n_k^m = n_k^{m-1}$ and $C_k^m = C_k^{m-1}$ for $k = 1, \ldots, N_R$. For $i = 1, \ldots, N_S$, all voxels in an environment of Γ_i^m are subsequently considered. Let a voxel vox be assigned to phase $k \in \{k^+(i), k^-(i)\}$ and let $l \neq k$ be the former phase index of the voxel. Then, we set

$$n_k^m = n_k^m + 1, \quad n_l^m = n_l^m - 1, \quad C_k^m = C_k^m + u_0|_{vox}, \quad C_l^m = C_l^m - u_0|_{vox}. \tag{5.30}$$

After having considered all voxels close to the surfaces, the coefficients are set to $c_k^m = C_k^m / n_k^m$ for $k = 1, \ldots, N_R$.

Time Step Control

We use a certain adaptive time step setting to control the speed of the evolution of the surface(s). Let $\Delta t = \tau_m$ denote the (possibly variable) time step size. The time step size is controlled as follows: Let $\delta X_n^{\min} > 0$, $\delta X_n^{\max} > 0$ with $\delta X_n^{\min} < \delta X_n^{\max}$ be user-defined tolerances for the absolute value of the position difference in normal direction. Let $\Delta t > 0$ be an initial time step size for $m = 0$ or the time step size of the previous time step for $m > 0$.

We propose the following time step size control: Choose a factor $\lambda_t \in \mathbb{N}$ (for example $\lambda_t = 2$ or $\lambda_t = 10$).

(i) Solve equation (5.28b) and set $\delta X_n^{m+1} := \max_{i=1,\ldots,N_S, j=1,\ldots,N_{i,V}} |\delta \vec{X}_{i,j}^{m+1} \,.\, \vec{\omega}_{i,j}^m|$.

(ii) If $\delta X_n^{m+1} > \delta X_n^{\max}$, set Δt to $\frac{1}{\lambda_t} \Delta t$ and repeat step (i).

(iii) Otherwise, if $\delta X_n^{m+1} < \delta X_n^{\min}$, set Δt to $\lambda_t \Delta t$ and repeat step (i).

(iv) Otherwise, proceed by checking for topology changes (see above) and go to the next time step, i.e. set m to $m + 1$.

The effect of this time step size control is simple: If there are too high changes in the position of the nodes in normal direction (i.e. if the normal velocity is too high), the time step size will be decreased. For mean curvature flow, this occurs if the surface has a (local) high curvature. For segmentation of 3D images, this occurs if the sum of weighted curvature and external term is high. If the change in the position in normal direction is too small, the time step size will be increased to speed up the image segmentation process.

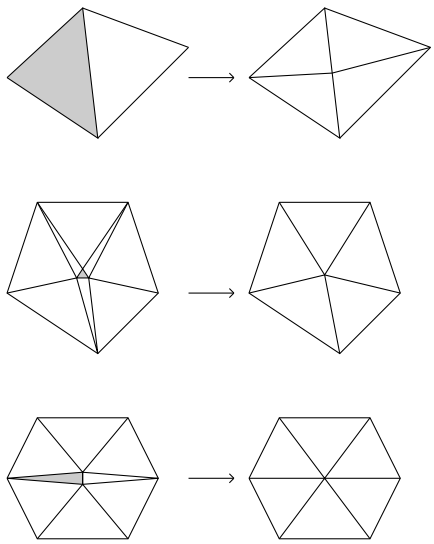


Figure 5.5: Top: Refinement of a simplex (marked in gray). The neighbor simplex is also refined to avoid hanging nodes. Center: Deletion of a simplex with a too small area (marked in gray). Three neighbor simplices are also deleted. Bottom: Deletion of a simplex with a too small angle (marked in gray). One neighbor simplex is also deleted.

Refinement and Deletion of Simplices

For computing the matrix entries, cf. (5.25), we already need to compute the area of each simplex of the triangulation of the surface Γ_i^m, $i \in \{1, \dots, N_S\}$. Let $A_{\text{desired}} > 0$ be a predefined desired area for one simplex. Let $a > 0$ be a given factor (e.g. $a = 2$ or $a = 10$). If the area of a simplex exceeds aA_{desired}, it will be refined by bisection of its largest edge. Its neighbor simplex across the refinement edge will be also refined such that no hanging nodes remain.

Local refinement is necessary to avoid too large simplices. Further, a mesh can also be continuously refined, for example when one starts with a small surface which globally grows. The triangles of the growing surface are refined one by one. Furthermore, the triangles which have one very large angle, i.e. an angle larger than a given threshold (e.g. $\geq 160^\circ$), are also refined.

If a simplex area is smaller than a certain percentage of the desired area A_{desired}, for example smaller than 1%, the simplex is deleted. Further, it is also deleted if one of its three inner angles is smaller than a given threshold, for example smaller than 2°. When a simplex is marked for deletion, one or more simplices are also deleted.

Mesh operations like deletion of triangles are rarely necessary. These operations are performed only a few times, for example close before or after topology changes.

Figure 5.5 illustrates examples how the triangulation is adapted close to simplices which are marked for refinement or deletion.

5.3.5 Mesh Regularization via Induced Tangential Motion

Applying the method of Barrett et al. (2008b) provides a good mesh quality in many cases. However, if, for example, a pinch-off occurs, a technique to avoid mesh distortion is needed. We will demonstrate the need of mesh regularization by an experiment in Section 5.4.

The idea of a mesh regularization method proposed by Barrett et al. (2008c) is to induce or reduce the tangential motion of nodes along a surface. We now use this method to control the tangential motion of nodes of surfaces during 3D image segmentation. According to Barrett et al. (2008c), we replace the system (5.19) by the following system:

Find $\{\kappa^{m+1}, \beta_1^{m+1}, \beta_2^{m+1}, \delta\vec{X}^{m+1}\} \in [W(\Gamma^m)]^3 \times \underline{V}(\Gamma^m)$, $m = 0, 1, \ldots, M-1$, such that for all $\chi \in W(\Gamma^m)$ and $\vec{\eta} \in \underline{V}(\Gamma^m)$

$$\tau_m \sigma \langle \kappa^{m+1}, \chi \rangle_m^h - \langle \delta\vec{X}^{m+1}, \chi\, \vec{\nu}^m \rangle_m^h = -\tau_m \langle F^m, \chi \rangle_m^h, \tag{5.31a}$$

$$\tau_m \langle \alpha_1^m (\delta_1^m \beta_1^{m+1} + c_1^m), \chi \rangle_m^h - \langle \alpha_1^m \delta\vec{X}^{m+1}, \chi \vec{\tau}_1^m \rangle_m^h = 0, \tag{5.31b}$$

$$\tau_m \langle \alpha_2^m (\delta_2^m \beta_2^{m+1} + c_2^m), \chi \rangle_m^h - \langle \alpha_2^m \delta\vec{X}^{m+1}, \chi \vec{\tau}_2^m \rangle_m^h = 0, \tag{5.31c}$$

$$\langle \kappa^{m+1}\, \vec{\nu}^m, \vec{\eta} \rangle_m^h + \sum_{k=1}^{2} \langle \alpha_k^m \beta_k^{m+1} \vec{\tau}_k^m, \vec{\eta} \rangle_m^h + \langle \nabla_s \delta\vec{X}^{m+1}, \nabla_s \vec{\eta} \rangle_m = -\langle \nabla_s \vec{X}^m, \nabla_s \vec{\eta} \rangle_m \tag{5.31d}$$

holds. In the equations above, $\vec{\tau}_1^m = (\vec{\tau}_{1,1}^m, \ldots, \vec{\tau}_{1,N_S}^m) \in \underline{V}(\Gamma^m)$ with $\vec{\tau}_{1,i}^m = \sum_{j=1}^{N_{i,V}} \vec{\tau}_{1,i,j}^m \chi_{i,j}^m$ and similarly $\vec{\tau}_2^m = (\vec{\tau}_{2,1}^m, \ldots, \vec{\tau}_{2,N_S}^m) \in \underline{V}(\Gamma^m)$ with $\vec{\tau}_{2,i}^m = \sum_{j=1}^{N_{i,V}} \vec{\tau}_{2,i,j}^m \chi_{i,j}^m$, such that $\{\vec{v}_{i,j}^m, \vec{\tau}_{1,i,j}^m, \vec{\tau}_{2,i,j}^m\}$ is an orthonormal basis of $\mathbb{R}^3$, for $i = 1, \ldots, N_S$ and $j = 1, \ldots, N_{i,V}$. Recall the definition of $\vec{v}_{i,j}^m = \vec{\omega}_{i,j}^m / \|\vec{\omega}_{i,j}^m\|$, cf. Section 5.3.1. Furthermore, $\alpha_1^m, \alpha_2^m, \delta_1^m, \delta_2^m \in W(\Gamma^m)$ are given coefficient vectors with $\alpha_{1,i,j} \geq 0$, $\alpha_{2,i,j} \geq 0$ and $c_1^m, c_2^m \in W(\Gamma^m)$ are given forcing terms.

We now consider a certain strategy for choosing the coefficient vectors and forcing terms proposed in Barrett et al. (2008c):

We consider the case

$$\alpha_1^m = \alpha_2^m \equiv \alpha \in \mathbb{R}, \quad \alpha \geq 0, \quad \delta_1^m = \delta_2^m \equiv 1, \quad c_1^m = c_2^m \equiv 0. \tag{5.32}$$

For $\alpha = 0$, the system reduces to (5.19). For $\alpha > 0$, we define $\vec{T}_{m,1}, \vec{T}_{m,2} \in (\mathbb{R}^3)^{N \times N}$ by

$$\vec{T}_{m,1} := \begin{pmatrix} \vec{T}_{m,1}^1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \vec{T}_{m,1}^{N_S} \end{pmatrix}, \quad \vec{T}_{m,2} := \begin{pmatrix} \vec{T}_{m,2}^1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \ldots & \vec{T}_{m,2}^{N_S} \end{pmatrix},$$

where

$$[\vec{T}_{m,1}^i]_{kl} := \langle \chi_{i,k}^m, \chi_{i,l}^m \vec{\tau}_1^m \rangle_m^h, \quad [\vec{T}_{m,2}^i]_{kl} := \langle \chi_{i,k}^m, \chi_{i,l}^m \vec{\tau}_2^m \rangle_m^h, \tag{5.33}$$

for $i = 1, \ldots, N_S$ and $k, l = 1, \ldots, N_{i,V}$. Again we identify elements in $\underline{V}(\Gamma^m)$ as elements in $(\mathbb{R}^3)^N$ and elements in $W(\Gamma^m)$ as elements in $\mathbb{R}^N$. From (5.31) we obtain the following linear system for $\delta\vec{X}^{m+1} \in (\mathbb{R}^3)^N$ and $\kappa^{m+1}, \beta_1^{m+1}, \beta_2^{m+1} \in \mathbb{R}^N$ using strategy (5.32):

$$\begin{pmatrix} \sigma\tau_m M_m & 0 & 0 & -\vec{N}_m^T \\ 0 & \tau_m M_m & 0 & -\vec{T}_{m,1}^T \\ 0 & 0 & \tau_m M_m & -\vec{T}_{m,2}^T \\ \vec{N}_m & \alpha\vec{T}_{m,1} & \alpha\vec{T}_{m,2} & \vec{A}_m \end{pmatrix} \begin{pmatrix} \kappa^{m+1} \\ \beta_1^{m+1} \\ \beta_2^{m+1} \\ \delta\vec{X}^{m+1} \end{pmatrix} = \begin{pmatrix} -\tau_m b_m \\ 0 \\ 0 \\ -\vec{A}_m \vec{X}^m \end{pmatrix}. \tag{5.34}$$

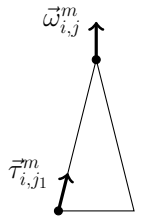


Figure 5.6: Illustration of a cone-like mesh after a splitting. The weighted normal at the peak nearly points in the same direction as tangent vectors at neighbor nodes. The weighted normal vector field is strongly discontinuous for such surfaces.

Here, we used the matrices M_m, $\vec{N}_m$, $\vec{A}_m$ and the vector b_m defined in (5.25) and (5.26).

Applying a Schur complement approach leads to

$$\kappa^{m+1} = \frac{1}{\sigma\tau_m} M_m^{-1} \vec{N}_m^T \delta\vec{X}^{m+1} - \frac{1}{\sigma} M_m^{-1} b_m, \tag{5.35a}$$

$$\beta_1^{m+1} = \frac{1}{\tau_m} M_m^{-1} \vec{T}_{m,1}^T \delta\vec{X}^{m+1}, \tag{5.35b}$$

$$\beta_2^{m+1} = \frac{1}{\tau_m} M_m^{-1} \vec{T}_{m,2}^T \delta\vec{X}^{m+1}, \tag{5.35c}$$

$$\left(\frac{1}{\sigma\tau_m} \vec{N}_m M_m^{-1} \vec{N}_m^T + \frac{\alpha}{\tau_m} \sum_{k=1}^{2} \vec{T}_{m,k} M_m^{-1} \vec{T}_{m,k}^T + \vec{A}_m \right) \delta\vec{X}^{m+1} = -\vec{A}_m \vec{X}^m + \frac{1}{\sigma} \vec{N}_m M_m^{-1} b_m. \tag{5.35d}$$

For $\alpha = 0$, (5.35d) reduces to (5.28b).

In practice, we only solve (5.35d) and do not explicitly compute $\kappa^{m+1}, \beta_1^{m+1}$ and β_2^{m+1}. Compared to (5.28b), the system matrix has to be slightly modified. We additionally compute $\vec{T}_{m,1} M_m^{-1} \vec{T}_{m,1}^T$ and $\vec{T}_{m,2} M_m^{-1} \vec{T}_{m,2}^T$.

For our applications, we slightly modified the strategy of Barrett et al. (2008c). Our modification is based on the following observation: If a surface is split into two new surfaces (after the detection of a pinch-off, for example), the two new surfaces are cone-like close to the former splitting point. The weighted normal vector field is strongly discontinuous at those parts of the surfaces. Let $\vec{X}_{i,j}^m$ be a node at the peak of the cone-like part of one of the two surfaces. We can observe that $\vec{\omega}_{i,j}^m$ nearly points in the direction of tangential vectors to the surface at neighbor nodes. This is illustrated in Figure 5.6.

Consequently, we perform the following modification: Instead of a global parameter α, we make use of a coefficient vector $\alpha^m \in \mathbb{R}^N$. For each node $\vec{X}_{i,j}^m$, $i = 1, \ldots, N_S$, $j = 1, \ldots, N_{i,V}$, we first set $\alpha_{i,j}^m = \alpha_0$ for a given default value $\alpha_0 > 0$. Then, we compare the direction of $\vec{\omega}_{i,j}^m$ with the direction of $\vec{\omega}_{i,j_1}^m$ for each neighbor node $\vec{X}_{i,j_1}^m$. (A neighbor node on the surface is a node $\vec{X}_{i,j_1}^m$, such that $\vec{X}_{i,j}^m$ and $\vec{X}_{i,j_1}^m$ are the endpoints of an edge of the triangulation of Γ_i^m.) If the angle between the two vectors $\vec{\omega}_{i,j}^m$ and $\vec{\omega}_{i,j_1}^m$ is larger than a threshold $\phi_{\max}$ (e.g. $\phi_{\max} = 20°$), we re-set $\alpha_{i,j}^m = \alpha_{i,j_1}^m = 0$.

In Barrett et al. (2008c) also other strategies are considered. The strategy (5.32) corresponds to strategy (i) in Barrett et al. (2008c). Here, we further apply our slight modification as described above. For the flows designed for image segmentation, this strategy is sufficient, cf. experiments in Section 5.4. Other strategies which involve non-zero forcing terms c_1^m and c_2^m are not needed for our applications.

A different routine for mesh regularization is developed by Bänsch et al. (2005). The main idea of their mesh regularization method is that all angles of triangles at one node should be of approximately the same size. For example, if a node belongs to n triangles, the angle in each triangle at the node should be $(2\pi)/n$. The mesh regularization routine in Bänsch et al. (2005) is not included in the linear system in contrast to Barrett et al. (2008c). The routine of Bänsch et al. (2005) is executed separately after the linear system is solved. For each node $\vec{X}_{i,j}^m$, the coordinates of the point $\vec{X}_{i,j}^m$ are first set to the average of $\vec{X}_{i,j}^m$ and all neighbor vertices. Second, the point is shifted in direction $\pm\vec{\omega}_{i,j}^m$ such that the enclosed volume is preserved. See Bänsch et al. (2005) for more details on the algorithm.

Chen (2004) considers several mesh smoothing methods based on so-called *optimal Delaunay triangulations*. The smoothing is based on minimization of the interpolation error among all triangulations with the same number of vertices. The methods distribute the edge length equally with respect to special metrics which use certain modifications of the Hessian matrix. Thus, also anisotropic problems can be handled. Three-dimensional examples are presented in Chen and Holst (2011). Applying such mesh smoothing methods to surface evolution problems results in executing a sub-algorithm for mesh smoothing in each iteration, or at least periodically, for example, after every ten time steps.

Here, we implemented a slight modification of the method of Barrett et al. (2008c) since it does not result in an additional routine. The tangential distribution of the nodes along the surface is directly included in the linear system that we have to solve.

For an alternative method for a tangential redistribution of mesh points, we also refer to Mikula et al. (2014b). The authors propose a volume-oriented and a length oriented tangential redistribution of mesh points on an evolving manifold. This is achieved by designing a special tangential velocity. Results from mean curvature flow and 3D image segmentation with well distributed mesh points are presented.

5.3.6 Area and Volume Computation

After the segmentation of a 3D image, one can compute the area of the surfaces and the volume of the enclosed regions. The regions typically represent certain objects of interest such as organs or tumors in medical images. As postprocessing, we like to compute the volume of the objects and the area of the boundaries of the objects.

Let $\Gamma^h \subset \Omega$, be a closed, oriented, triangulated surface, and let $\Omega_0^h \subset \Omega$ be a three-dimensional subset such that $\partial\Omega_0^h = \Gamma^h = \bigcup_{j=1}^{N_F} \sigma_j^h$, where σ_j^h denote the triangles of the triangulation, $j = 1, \ldots, N_F$. The area of Γ^h enclosing the region Ω_0^h is

$$|\Gamma^h| = \sum_{j=1}^{N_F} |\sigma_j^h|, \tag{5.36}$$

where $|\sigma_j^h|$ denotes the area of the triangle σ_j^h.

For the computation of the volume of Ω_0^h, we use a divergence theorem. Let $\vec{\nu}_j^h = \vec{\nu}^h|_{\sigma_j^h} \in \mathbb{R}^3$ be the outer normal vector on the simplex σ_j^h, $j = 1, \ldots, N_F$, and let $\vec{\nu}^h$ be a normal vector field on Γ^h with $\vec{\nu}^h|_{\sigma_j^h} = \vec{\nu}_j^h$. Since the boundary of Ω_0^h is piecewise smooth, (it is non-smooth only at the edges of the triangles), the volume can be computed as follows:

$$\mathrm{vol}(\Omega_0^h) = \int_{\Omega_0^h} 1 \,\mathrm{d}x = \frac{1}{3} \int_{\Omega_0^h} \mathrm{div}(\vec{x}) \,\mathrm{d}x = \frac{1}{3} \int_{\Gamma^h} \vec{x} \,.\, \vec{\nu}^h \,\mathrm{d}A. \tag{5.37}$$

Since Γ^h is composed of triangles and since $\vec{x} \,.\, \vec{\nu}^h$ is constant on each triangle, the expression above can be written to

$$\mathrm{vol}(\Omega_0^h) = \frac{1}{3} \sum_{j=1}^{N_F} \int_{\sigma_j^h} \vec{x} \,.\, \vec{\nu}_j^h \,\mathrm{d}A = \frac{1}{3} \sum_{j=1}^{N_F} |\sigma_j^h| \, \vec{x}_j^h \,.\, \vec{\nu}_j^h. \tag{5.38}$$

Here, $\vec{x}_j^h$ is an arbitrary representative point of the triangle σ_j^h (for example one of the three vertices).

As mentioned above, the computation of the area of the surfaces and the volume of the regions enclosed by the surfaces is typically done at the end of the segmentation. Let Γ_i^M, $i = 1, \ldots, N_S$, denote the final surfaces. Then, we apply the computation described in this section on each surface $\Gamma^h = \Gamma_i^M$, $i \in \{1, \ldots, N_S\}$, separately.

5.3.7 Summary of the Image Segmentation Algorithm

In summing up, we propose the following algorithm for segmentation of 3D images. Given a set of triangulated surfaces $\Gamma^0 = (\Gamma_1^0, \ldots, \Gamma_{N_S}^0)$ and nodes $\vec{X}_{i,j}^0$, $i = 1, \ldots, N_S$, $j = 1, \ldots, N_{i,V}$, perform the following steps for $m = 0, 1, \ldots, M-1$:

(i) Compute the regions Ω_k^m and the coefficients c_k^m, $k = 1, \ldots, N_R$, as described in Section 5.3.4.

(ii) Compute b^m as defined in (5.26) by using the coefficients c_k^m of step (i). Compute the coefficient vector α^m as described in Section 5.3.5. Compute $\vec{X}^{m+1} = \vec{X}^m + \delta\vec{X}^{m+1}$ by solving the linear equation (5.35d), see Section 5.3.5.

(iii) Check whether the time step size needs to be increased or decreased, see Section 5.3.4. If the time step size needs to be changed, repeat step (ii) with the new time step size.

(iv) Check whether topology changes occur and execute the topology change, see Section 5.3.3.

(v) If necessary, refine too large simplices or delete too small simplices of the triangulation as described in Section 5.3.4.

As postprocessing, one can compute the volume of the enclosed regions and the area of the surfaces as described in Section 5.3.6.

In this chapter, we have not described how 3D images can be denoised. In principle, the concept for image denoising with edge enhancement of the two-dimensional case could be generalized to the three-dimensional case. Diffusion equations with Neumann boundary conditions similar to (3.43) can be solved numerically with a finite difference approach. Instead of the 2D pixel grid, we could use the 3D voxel grid.

5.4 Results

5.4.1 Mean Curvature Flow and Artificial Test Images

In this section we demonstrate the evolution of surfaces with topology changes. We present two examples of mean curvature flow of a surface where the topology changes splitting and

decrease of genus occur. We further present examples from image segmentation to demonstrate splitting, merging, increase and decrease of genus. Furthermore, we show the necessity of a mesh regularization technique like the induced tangential motion of Barrett et al. (2008c) (cf. Section 5.3.5).

Splitting examples

Figure 5.7 shows an example where a surface evolves under mean curvature flow. Our method for image segmentation reduces to mean curvature flow when setting $\sigma = 1$ and $\lambda = 0$ (cf. evolution equation (5.5) and definition of the external forcing term (5.4)).

Note that the initial surface in Figure 5.7 is non-convex. Grayson (1987) showed that a smooth, embedded *curve* which evolves under mean curvature flow shrinks to a point, and its limiting shape is a round circle. For higher dimensions, the analog statement is true for *convex hypersurfaces*, see Huisken (1984). In the example, which we consider in Figure 5.7, the non-convex surface develops a singularity (pinch-off). Here, we like to proceed, i.e. we want to perform a topology change and we let the surface split into two single surfaces.

To detect the topology change, we use an auxiliary background grid with grid size $a = 0.025$ as described in Section 5.3.3. A cube of the grid will be considered for possible topology changes if more than $N_{\text{detect}} = 10$ nodes are located inside the cube. The topology change is identified as a splitting at time step $m = 69$ as described in Section 5.3.3, where we use the parameters $thr1 = 30°$, $thr2 = 150°$ and $thr3 = 40°$.

Furthermore, we also perform a time step control (cf. Section 5.3.4) using the thresholds $\delta X_n^{\min} = 0.003$ and $\delta X_n^{\max} = 0.05$. After the splitting, the surfaces retract fast close to the former splitting point due to the local high curvature. The time step size Δt is reduced from 10^{-3} to 10^{-5} to prevent a too large retraction from one iteration step to the next. After a few time steps Δt is increased intermediately to 10^{-4} and further increased to 10^{-3} for steps larger than $m = 125$. At the end, each of the two surfaces shrinks to a point. The time step size is decreased a second time. The iteration is stopped at $m = 160$ due to the too small area of the surfaces. The time step sizes with respect to the iteration step are plotted in Figure 5.8.

For this example, we use a mesh regularization via induced tangential motion, see Section 5.3.5. We apply strategy (5.32), see also Barrett et al. (2008c), with our slight modification. We set $\alpha_0 = 10$ to control the induced tangential motion. Nodes $\vec{X}_{i,j}^m$, where the discrete normal vector $\vec{\omega}_{i,j}^m$ is strongly discontinuous, are not affected by the induced tangential motion. Figure 5.9 shows a magnification of the surface at time step $m = 70$. For nodes $\vec{X}_{i,j}^m$ belonging to the green colored parts of the surface, a tangential motion is enforced and $\alpha_{i,j}^m = \alpha_0 = 10$ is used. For nodes belonging to the red colored parts, $\alpha_{i,j}^m = 0$ is used to suppress an additional tangential motion.

The proposed modification is necessary. Figure 5.10 shows the resulting surface shortly after the topology change if a global parameter α is used instead. The peak close to the former splitting location does not retract. This is a numerical problem: Analytically, a surface, which evolves under mean curvature flow, quickly retracts at locations with high local curvature. The discontinuous weighted normal vector $\vec{\omega}_{i,j}^m$ probably causes this problem. This kind of numerical artifact does no longer occur when setting $\alpha_{i,j}^m = 0$ at such peaks, cf. Section 5.3.5.

Mesh regularization is important in extreme situations during surface evolution like the formation of a pinch-off. We repeat the experiment without any mesh regularization. After just a few iterations steps the mesh gets heavily distorted, see Figure 5.11.

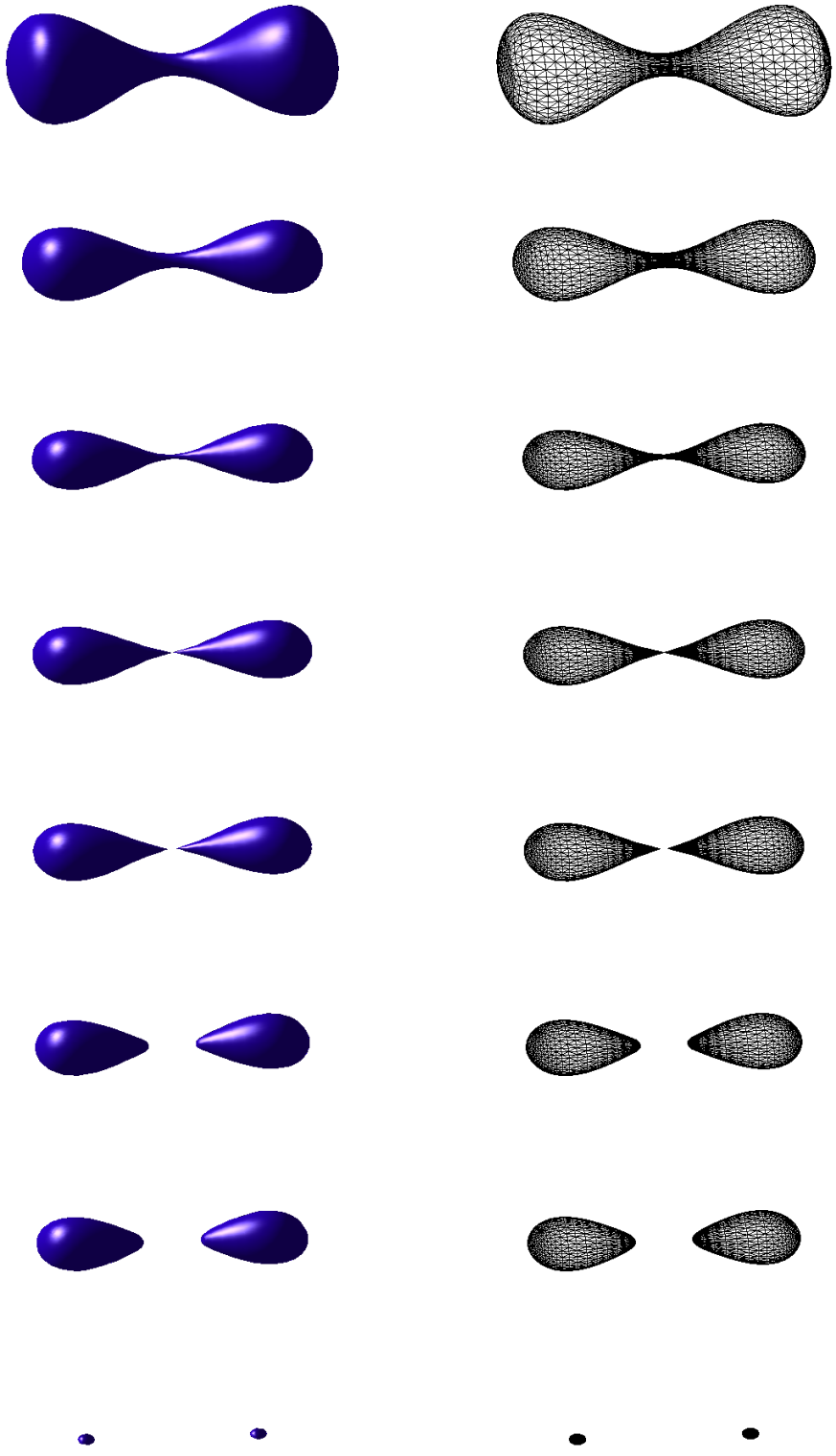


Figure 5.7: Mean curvature flow with pinch-off and splitting into two surfaces. Surfaces (left) and meshes (right) at step $m = 0, 50, 68, 69, 70, 100, 120, 159$ (row-wise) at time $t_m = 0, 0.050, 0.068, 0.069, 0.070, 0.0727, 0.0747, 0.1083$.

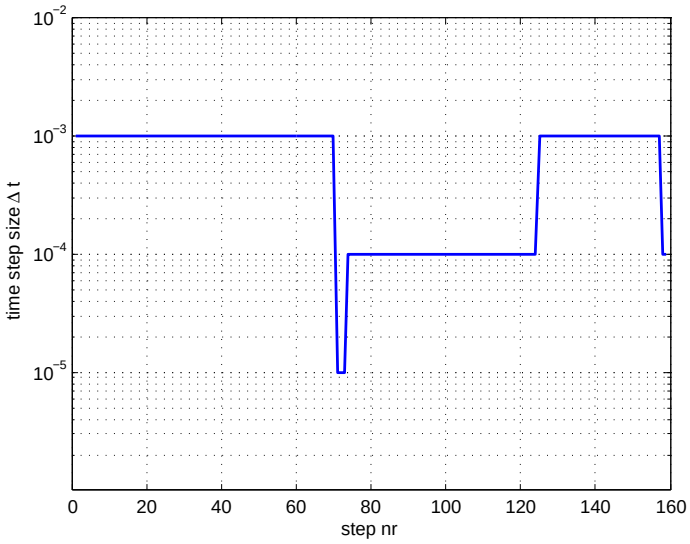


Figure 5.8: Time step sizes during the evolution of the surface. After the splitting the time step size is decreased.

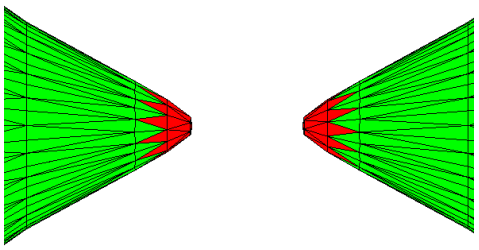


Figure 5.9: Magnification at time step $m = 70$ close to the former splitting location. Red parts: $\alpha_{i,j}^m = 0$, green parts: $\alpha_{i,j}^m = 10$ (parameter controlling the induced tangential motion).

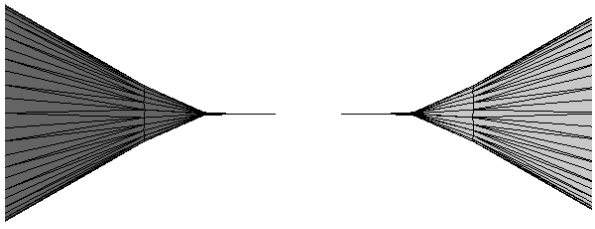


Figure 5.10: Illustration of the numerical artifact if a global value α is used for all nodes. We can observe that the surfaces do not retract. This is a numerical problem induced by the strongly discontinuous weighted normal vector $\vec{\omega}^m$. Such situations will not occur when using our proposed method, see Section 5.3.5.

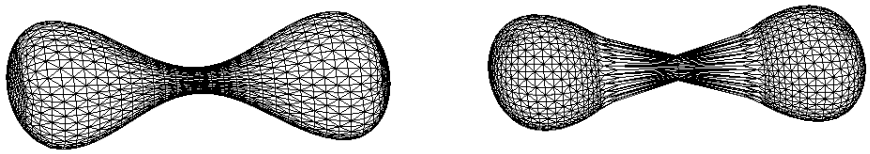


Figure 5.11: Mean curvature flow example without mesh regularization. Surface at step $m = 0$ and $m = 15$. This demonstrates the necessity of a mesh regularization technique.

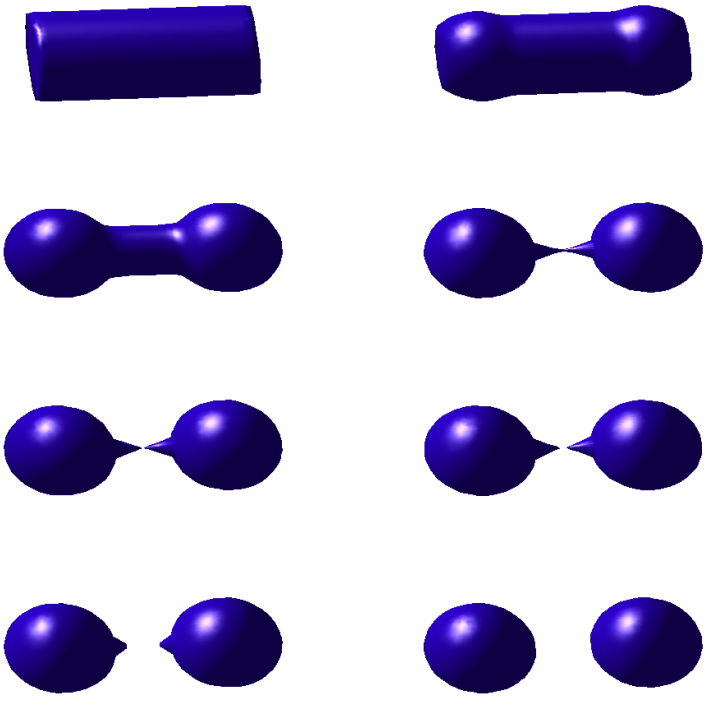


Figure 5.12: Splitting of a surface during 3D image segmentation. Surface(s) at step $m = 0, 25, 100, 213, 214, 215, 230, 300$ (row-wise) at time $t_m = 0, 0.025, 0.01, 0.0213, 0.0214, 0.0215, 0.0228, 0.0298$.

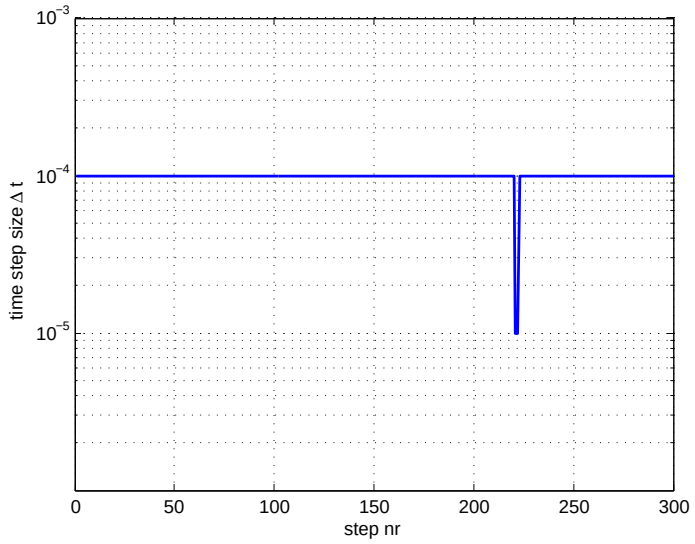


Figure 5.13: Time step sizes during the evolution of the surface. After the splitting the time step size is decreased for a few steps.

Another splitting example is shown in Figure 5.12. The initial surface is a cylinder-like surface. The image domain is given by the cuboid $\Omega = [-2.5, 2.5] \times [-1.5, 1.5] \times [-1.5, 1.5]$ and the image intensity function is defined by

$$u_0 : \Omega \to \mathbb{R}, \quad u_0(\vec{x}) = \begin{cases} 0 & \text{if } \|\vec{x} - (-1.2, 0, 0)^T\| \leq 0.8 \vee \|\vec{x} - (1.2, 0, 0)^T\| \leq 0.8, \\ 1 & \text{else.} \end{cases}$$

At time step $m = 214$ a splitting occurs. In the subsequent iterations steps, the two new surfaces each evolve to a ball with radius 0.8 centered at $(\pm 1.2, 0, 0)^T$. For weighting the curvature term and the forcing term for the image segmentation, the parameters $\sigma = 1$ and $\lambda = 100$ are used. For the induced tangential motion, the detection of the topology change and for the time step size control, the same parameters as for the mean curvature flow example above are applied. The time step sizes are visualized in Figure 5.13.

Merging example

The reversed topology change of splitting is a merging of two surfaces to one single surface. The initial surfaces in our next example are two balls. The image domain is given by $\Omega = [-1.2, 1.2] \times [-0.8, 0.8] \times [-0.8, 0.8]$ and the image intensity function is defined by

$$u_0 : \Omega \to \mathbb{R}, \quad u_0(\vec{x}) = \begin{cases} 0 & \text{if } \|\vec{x}\| \leq 0.6, \\ 1 & \text{else.} \end{cases}$$

As weighting parameters $\sigma = 2$ and $\lambda = 60$ are used; for the induced tangential motion method $\alpha_0 = 0.1$ is used. Time step control is performed applying the thresholds $\delta X_n^{\min} = 0.001$ and $\delta X_n^{\max} = 0.02$. No change of the time step size is necessary in this example; the time step size $\Delta t = 10^{-4}$ need not be changed throughout the evolution. To detect the merging, $a = 0.03$, $N_{\text{detect}} = 10$ and $thr1 = 20°$, $thr2 = 150°$ and $thr3 = 40°$ are used. The resulting surfaces of this experiment at different time steps are shown in Figure 5.14.

Since the surface grows continuously, some simplices have to be refined as described in Section 5.3.4. The desired area for one simplex is $A_{\text{desired}} = 0.001$; a simplex is refined by

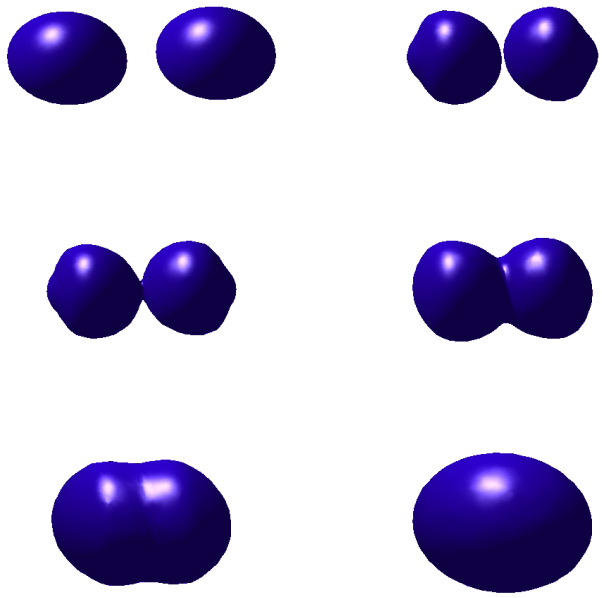


Figure 5.14: Merging demonstration. Surface(s) at step $m = 0, 49, 50, 80, 150, 200$ (row-wise) at time $t_m = 0, 0.0049, 0.005, 0.008, 0.015, 0.02$.

bisection of its largest angle if its area is larger than a certain factor of 0.001. A simplex is also bisected if one angle is larger than 170°. A simplex is deleted if one angle is smaller than 2° or if its area is smaller than 1% of the desired area of $A_{\text{desired}} = 0.001$.

Examples demonstrating increase and decrease of the genus

In the next examples, we demonstrate another kind of topology changes: increase and decrease of the genus of a surface. Therefore, we consider an example from image segmentation where a sphere should evolve to a torus. The image intensity function is given by

$$u_0(\vec{x}) = \begin{cases} 0 & \text{if } (\sqrt{x_1^2 + x_2^2} - R)^2 + x_3^2 \leq r^2, \\ 1 & \text{else,} \end{cases} \tag{5.39}$$

where $R = 1.2$ and $r = 0.4$ are used here.

Figure 5.15 shows the surface, its mesh and a cross-section of the mesh at different time steps. For this example we apply $\sigma = 1$, $\lambda = 60$ (weighting parameters) and $\alpha_0 = 0.1$ (parameter controlling the induced tangential motion). The topology change is detected using $a = 0.0565$, $N_{\text{detect}} = 8$ and $thr1 = 20°$, $thr2 = 150°$ and $thr3 = 40°$. As parameters to control the refinement, the desired triangle area is set to $A_{\text{desired}} = 0.005$, and the angles 170° and 2° are used for bisection or deletion of a triangle, respectively.

In this example, the initial mesh is generated by using a simple triangulated cube, followed by a projection of each vertex to the sphere. This results in an initial mesh with a relatively poor mesh quality, i.e. the vertices are not distributed equally over the surface. This example shows that our method can also cope with difficult meshes: Large triangles are bisected and our method distributes the nodes over the surface during the evolution of the surface.

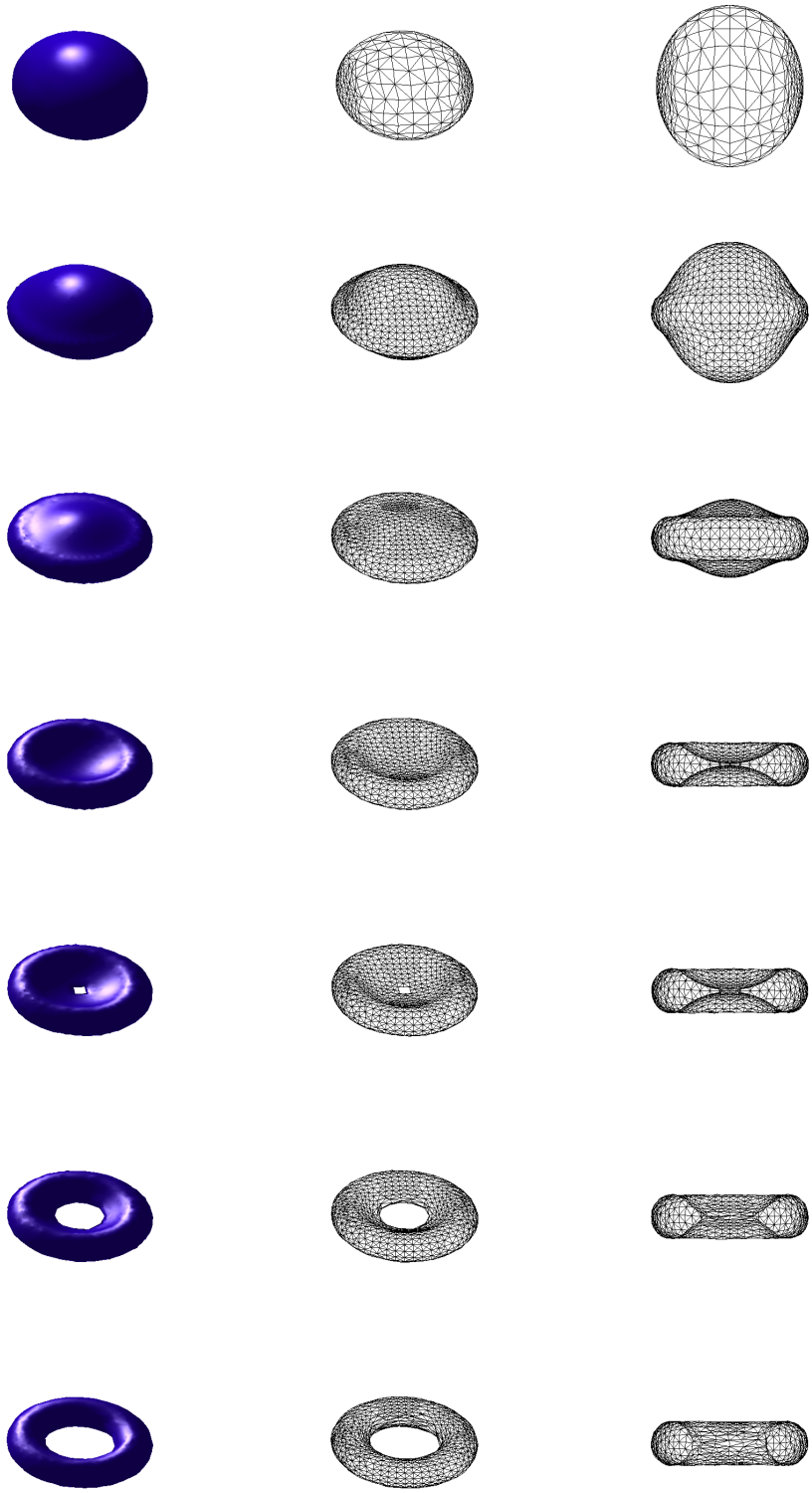


Figure 5.15: Demonstration of an increase of the genus of a surface. Surface, mesh and cross-section at step $m = 0, 25, 100, 325, 326, 380, 500$ (rows 1-6) at time $t_m = 0, 0.025, 0.1, 0.325, 0.326, 0.38, 0.5$. Column 1-3: surface, mesh, mesh (cross-section).

Further, $\delta X_n^{\min} = 0.01$ and $\delta X_n^{\max} = 0.1$ are applied as thresholds for the time step size control. Throughout the evolution, there was no need to change the initial time-step size of $\Delta t = 10^{-3}$.

At time step $m = 325$, before the topology change is detected, the mesh consists of $F = 3488$ simplices (faces), $V = 1746$ nodes (vertices) and $E = 5232$ edges, resulting in an Euler characteristic of $\chi = V - E + F = 2 = 2 - 2g$ for a surface of genus $g = 0$ (sphere-like surface). At time step $m = 326$, after the topology change has been executed, the mesh consists of $F = 3464$ simplices, $V = 1732$ nodes and $E = 5196$ edges, resulting in $\chi = V - E + F = 0 = 2 - 2g$ for a surface of genus $g = 1$ (torus-like surface).

Next, we demonstrate the evolution of a torus to a surface of genus 0, i.e. the genus is decreased. First, we consider an example where a torus evolves under mean curvature flow. Figure 5.16 shows the surface and the mesh from different viewing angles and a cross-section of the surface at several time steps. At time step $m = 25$ a topology change is detected. The triangulation of the surface is adapted as described in Section 5.3.3. For the detection of the decrease of genus, the parameters $a = 0.025$, $N_{\text{detect}} = 20$ and $thr1 = 20°$, $thr2 = 150°$ and $thr3 = 40°$ are used.

At time step $m = 24$ the mesh consists of $F = 2560$ faces, $V = 1280$ vertices and $E = 3840$ edges, resulting in an Euler characteristic $\chi = V - E + F = 0 = 2 - 2g$ for a surface of genus $g = 1$. At time step $m = 25$, after the topology change, the mesh consists of $F = 1960$ faces, $V = 982$ vertices and $E = 2940$ edges, resulting in $\chi = V - E + F = 2 = 2 - 2g$ for a surface of genus $g = 0$.

The time step size is controlled using the thresholds $\delta X_n^{\min} = 0.0005$ and $\delta X_n^{\max} = 0.01$. Figure 5.17 shows the time step sizes with respect to the iteration step. After the change of the topology of the surface, the time step size is decrease from 10^{-3} to 10^{-5}. After some steps, the size is increased to 10^{-4}.

We apply $\alpha_0 = 10$ to control the induced tangential motion. Thus, we can perform a mesh regularization. Mesh regularization is important in this situation. Figure 5.18 demonstrates the necessity of a mesh regularization technique. It shows how the mesh close to the hole can get strongly distorted.

We present an example from image segmentation where a torus is used as initial surface and a sphere should be detected. The image intensity function is given by

$$u_0 : \Omega \to \mathbb{R}, \quad u_0(\vec{x}) = \begin{cases} 0 & \text{if } \|\vec{x}\| \leq 0.8, \\ 1 & \text{else.} \end{cases}$$

Figure 5.19 shows the surface at several time steps. As weighting parameters $\sigma = 1$ and $\lambda = 20$ are applied. The tangential motion parameter is set to $\alpha_0 = 10$. We use the same parameters as in the mean curvature flow example to control the time steps size and to detect the change in topology. Figure 5.20 shows the time step sizes during the image segmentation process. After the topology change the time step size is decreased from 10^{-4} to 10^{-5}. Later it is increased to speed up the segmentation.

Multiple topology changes

Figure 5.21 demonstrates an example where more than one topology change occurs during the segmentation of an artificial 3D image. The image consists of $200 \times 100 \times 100$ voxels. We use $\sigma = 1$ and $\lambda = 50$ as weighting parameters. As above we set the tangential motion

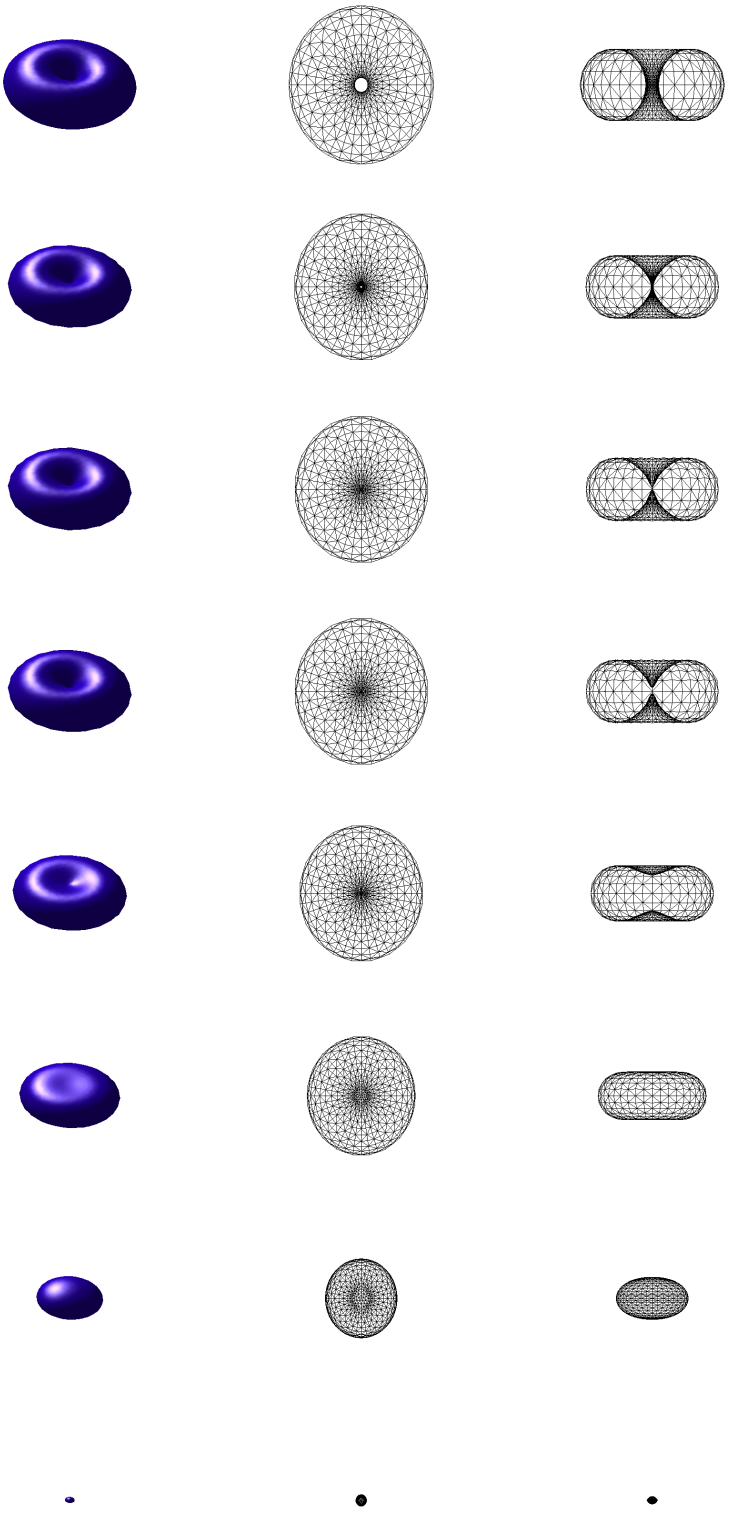


Figure 5.16: Mean curvature flow of a torus with decrease of its genus. Surface at step $m = 0, 24, 25, 50, 250, 500, 1000, 1480$ (rows 1-8) at time $t_m = 0, 0.024, 0.025, 0.02525, 0.04309, 0.06809, 0.11809, 0.16609$. Column 1-3: surface, mesh (top view), mesh (cross-section).

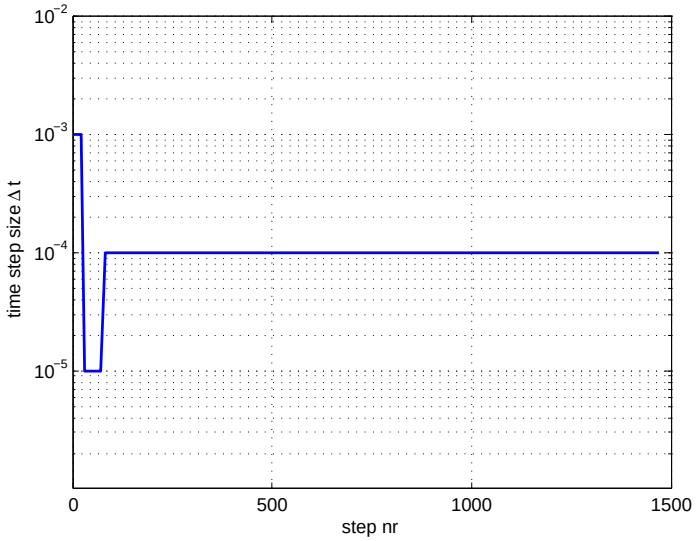


Figure 5.17: Time step sizes during the mean curvature flow of a torus.

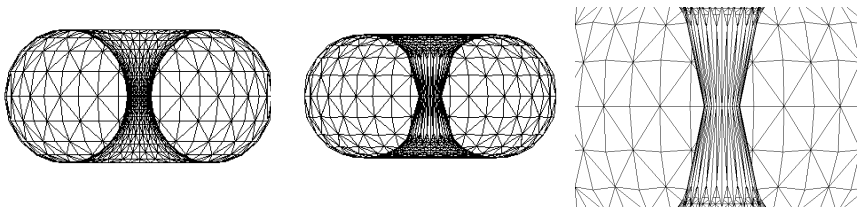


Figure 5.18: Mean curvature flow of the torus without mesh regularization. Cross-section of the mesh at step $m = 0$ (left) and $m = 40$ (center). Right: Magnification of the cross-section at step $m = 40$ to visualize the distorted mesh. This example demonstrates the necessity of a mesh regularization technique.

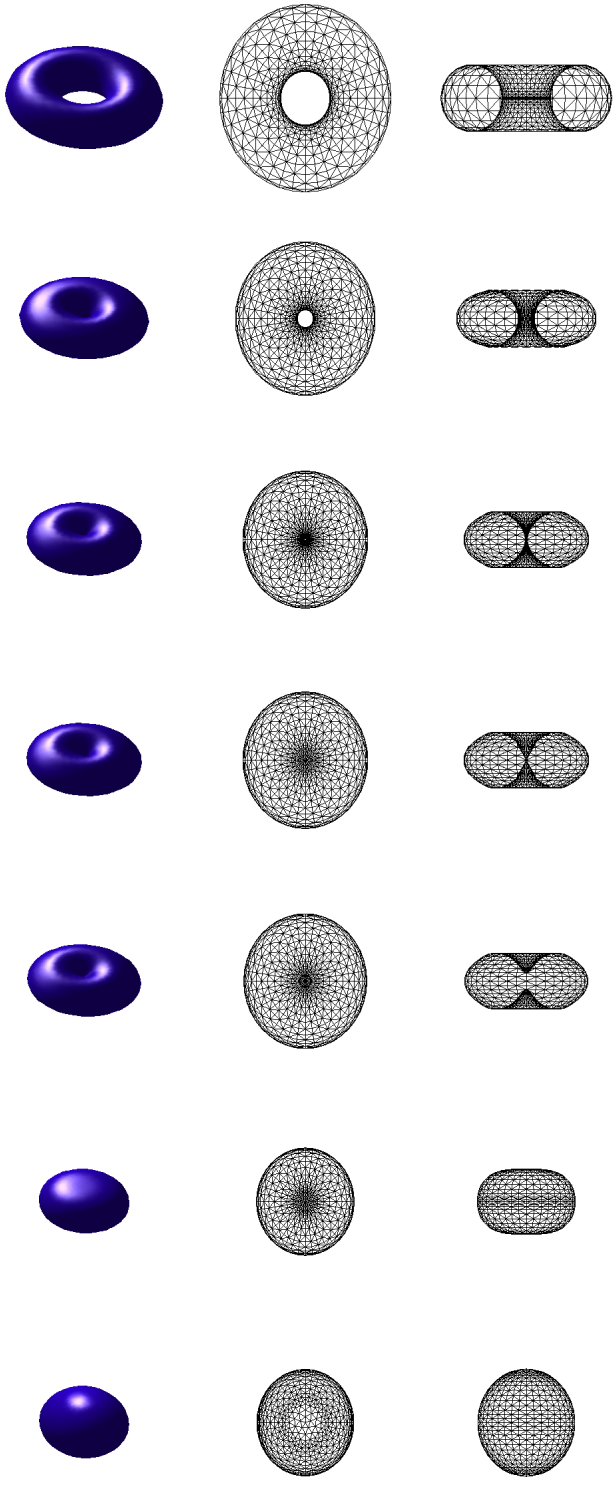


Figure 5.19: 3D image segmentation example where a torus evolves to a ball. Surface at step $m = 0, 300, 425, 426, 500, 750, 1000$ (rows 1-7) at time $t_m = 0, 0.03, 0.0425, 0.0426, 0.04406, 0.06906, 0.27316$. Column 1-3: surface, mesh (top view), mesh (cross-section).

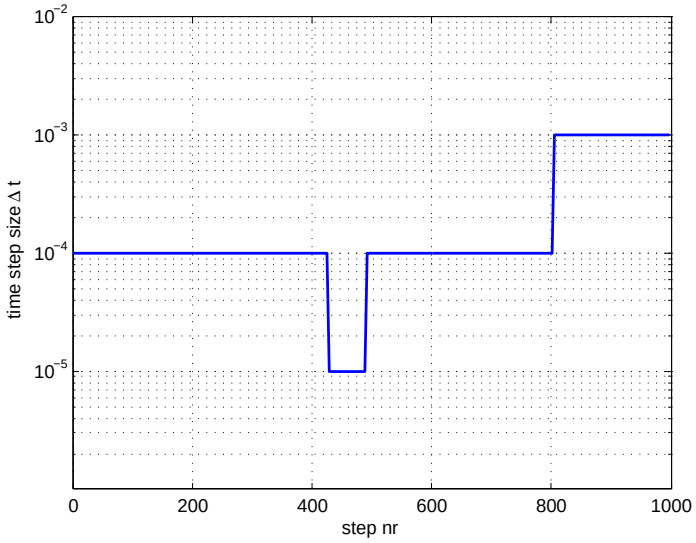


Figure 5.20: Time step sizes during the evolution of the torus to a ball.

parameter to $\alpha_0 = 10$. We use the thresholds $\delta X_n^{\min} = 0.05$ and $\delta X_n^{\max} = 1$ to control the time step size. The topology change is detected using $a = 1$, $N_{\text{detect}} = 5$ and $thr1 = 30°$, $thr2 = 150°$ and $thr3 = 40°$. As parameter to control the refinement, the desired triangle area is set to $A_{\text{desired}} = 1.5$.

In this example three topology changes occur: A splitting of the surface in two surfaces, an increase of the genus of one of the surfaces (evolution to a torus), and finally a second splitting. The final segmentation consists of a torus, a sphere and a cuboid. Figure 5.22 shows different cross-sections of the surface(s) and of the 3D image for some depicted time steps: at the beginning (1st row), after the first splitting (2nd row), after the increase of the genus of the left surface (3rd row) and the cross-sections of the final segmentation (4th row).

Computation of area and volume

In Section 5.3.6, we described how the area of the surfaces and the volume of the enclosed regions can be computed. We now demonstrate the computation of area and volume by considering triangulations of a ball with radius 1. Figure 5.23 shows three triangulations of a ball. The surfaces of the 2nd and 3rd subfigure are refinements of the surface shown in the 1st subfigure. The refinement is performed by bisection of the triangles followed by a projection of the vertices to the sphere, such that the norm of each vertex is 1. Table 5.1 shows the resulting values and errors of the computation of the area of the triangulations and of the volume of the enclosed regions. The row denoted with *Reference* presents the true values of the area and the volume of a ball with radius 1. Since the ball is convex and each vertex has norm 1, we face a negative error in the area and volume computations. The accuracy of the computations of the area and the volume improves with the number of refinement steps. Thus with decreasing size of the simplices, the area and the volume converges to the true values.

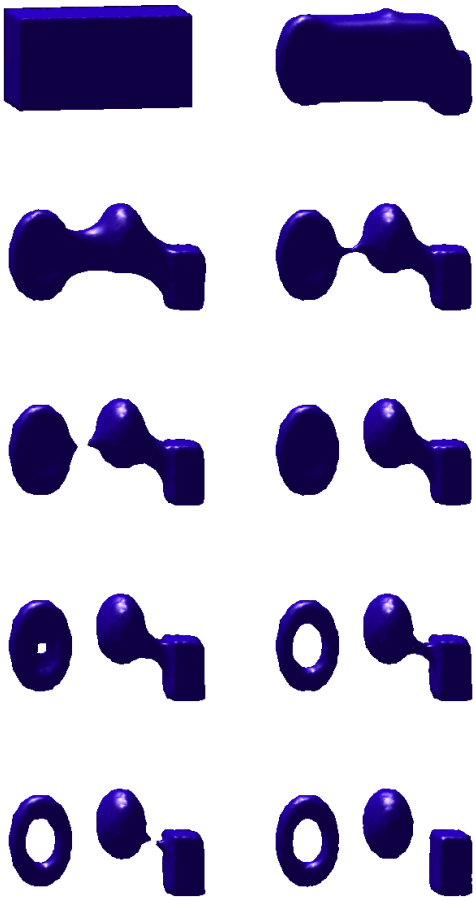


Figure 5.21: Detection of three objects. Surface(s) at step $m = 0, 100, 500, 1380, 1390, 1500, 1730, 2000, 2230, 2500$.

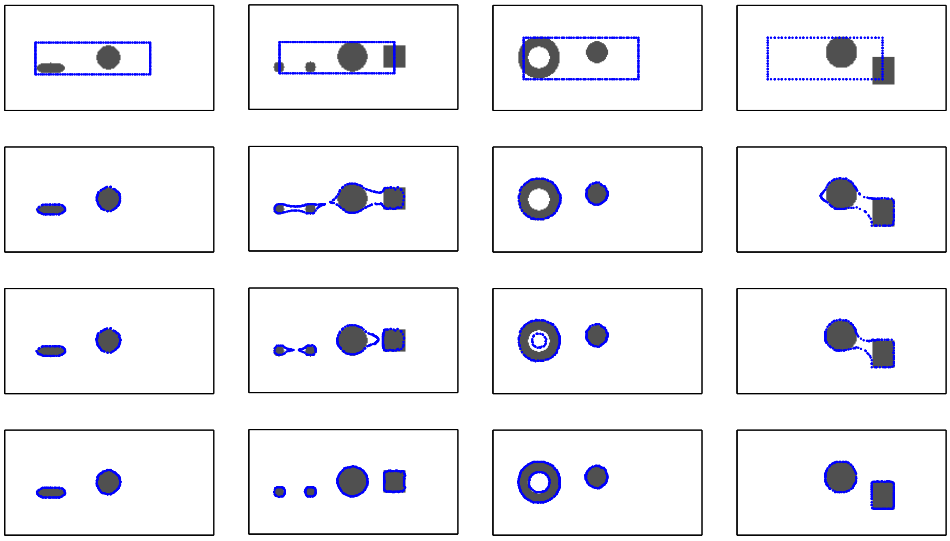


Figure 5.22: Detection of three objects - Cross sections of the image and the surface at step $m = 0, 1380, 1730, 2500$ (row 1-4). Cross-sections levels: $z = 35$ (column 1), $z = 50$ (column 2), $y = 40$ (column 3), $y = 50$ (column 4).

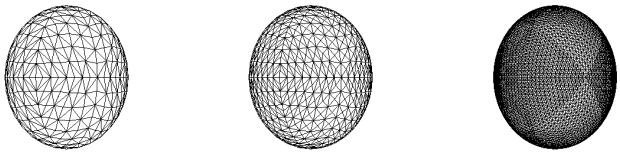


Figure 5.23: Triangulation of a ball, left: initial triangulation (768 simplices, 386 nodes), center: 1 refinement (1536 simplices, 770 nodes), right: 4 refinements (12288 simplices, 6146 nodes).

	Area	Relative Error Area	Volume	Relative Error Volume
Reference	12.566	0%	4.189	0%
Initial Triangulation	12.469	-0.78%	4.110	-1.87%
1 Refinement	12.516	-0.40%	4.149	-0.94%
4 Refinements	12.560	-0.05%	4.184	-0.12%

Table 5.1: Computation of the Area and Volume of a Ball with Radius 1

5.4.2 Segmentation of Medical 3D Images

In this section, we apply the segmentation method for three-dimensional images to medical image data. Segmentation of medical images is a challenging task due to possible high noise and image artifacts, see Sharma and Aggarwal (2010).

3D image data often consists of a set of 2D slice images generated by radiology scans, for example computed tomography (CT) and magnetic resonance (MR) scans. With a 3D image segmentation technique, one can segment organs (heart, lung, abdomen, liver, etc.) or tumors from their environment. The output, i.e. the resulting surface, serves as a reconstruction and visualization of the medical object and could be used for further medical analysis and diagnostic purposes: After the segmentation, one can compute the area of the triangulated surfaces. Further, also the volume of the enclosed regions can be determined in a postprocessing computation, see Section 5.3.6. The area of the surfaces and the volume of the regions could be used for example to analyze if a tumor has been growing in the time between two radiological examinations.

Example using data from The Cancer Imaging Archive

For a first experiment, we consider a sample 3D image of the Lung Image Database Consortium image collection (LIDC-IDRI) of The Cancer Imaging Archive (TCIA) (`https://wiki.cancerimagingarchive.net/display/Public/LIDC-IDRI`), see Reeves et al. (2007); Armato et al. (2011); Reeves and Biancardi (2011))[1]. The data set consists of diagnostic CT scans. The original data set consists of 2D slice images stored as DICOM files. The files are first preprocessed to cuboid 3D images with $N_x \times N_y \times N_z$ voxels, here: $N_x = 445$, $N_y = 310$ and $N_z = 250$.

Figures 5.24-5.27 show the evolving 3D surfaces and six representative 2D cross-sections at different time steps $m = 0, 100, 300, 600$. The surfaces are drawn from different viewing angles: the azimuth and elevation angles are $az = -5°$, $el = 45°$ (1st and 2nd subfigure, rowwise) and $az = 0°$, $el = 0°$ (3rd subfigure, front view) and $az = 0°$, $el = 90°$ (4th subfigure, top view).

In the second subfigure of Figures 5.24-5.27, those parts of the surfaces where the tangential motion is suppressed (i.e. $\alpha_{i,j}^m = 0$) are marked with red color. There, the change in the normals $\vec{v}_{i,j}^m = \vec{\omega}_{i,j}^m / \|\vec{\omega}_{i,j}^m\|$ at the vertices $\vec{X}_{i,j}^m$ is locally large, in detail larger than $\phi_{\max} = 50°$. At locations on the surface, where $\vec{v}_{i,j}^m$ is more continuous, a tangential motion is allowed and $\alpha_{i,j}^m = \alpha_0 = 0.1$ is used. These parts are drawn with green color.

In the subfigures showing 2D cross-sections, the image cross-sections for constant z (in detail $z = 80, 150, 200$) and constant y (in detail $y = 80, 150, 200$) are drawn as well as the intersection points of the surfaces' edges with the cross-section planes.

For the image segmentation the weight of the curvature term is set to $\sigma = 10$, the weight of the external forcing term to $\lambda = 1000$. As parameters for the time step size control, $\delta X_n^{\min} = 0.05$ and $\delta X_n^{\max} = 2$ are used. The time step size $\Delta t = 0.1$ need not be changed during the segmentation.

[1]The author acknowledges the National Cancer Institute and the Foundation for the National Institutes of Health, and their critical role in the creation of the free publicly available LIDC/IDRI Database.

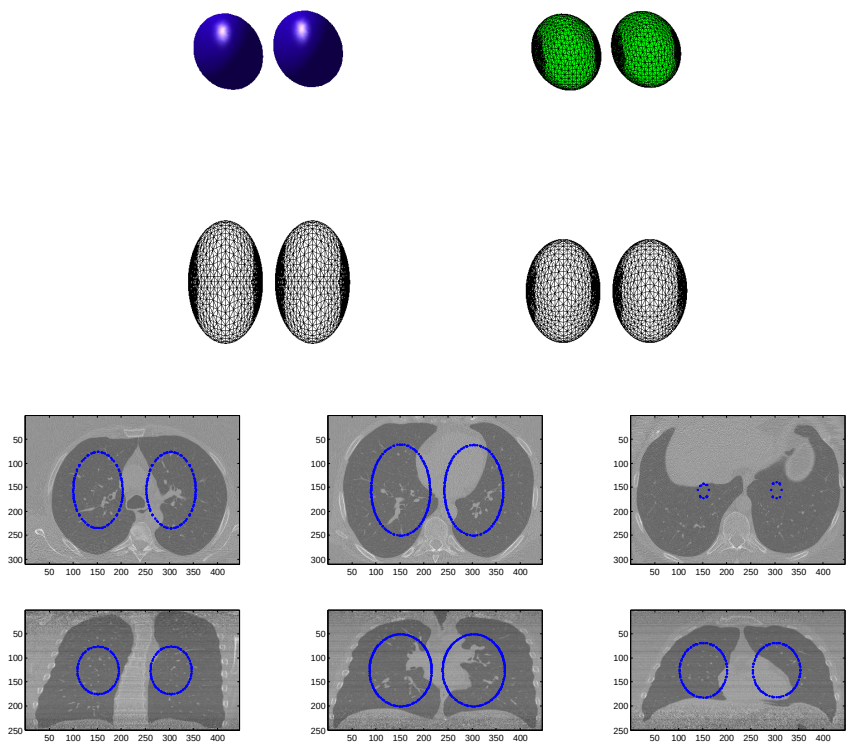


Figure 5.24: Lung segmentation: Surface at different viewing angles (row 1-2) and cross-sections (row 3: $z = 80, 150, 200$, row 4: $y = 80, 150, 200$) at $m = 0$ at time $t = 0$. The original images are from the Lung Image Database Consortium image collection (LIDC-IDRI) of The Cancer Imaging Archive (TCIA), see Reeves et al. (2007); Armato et al. (2011); Reeves and Biancardi (2011).

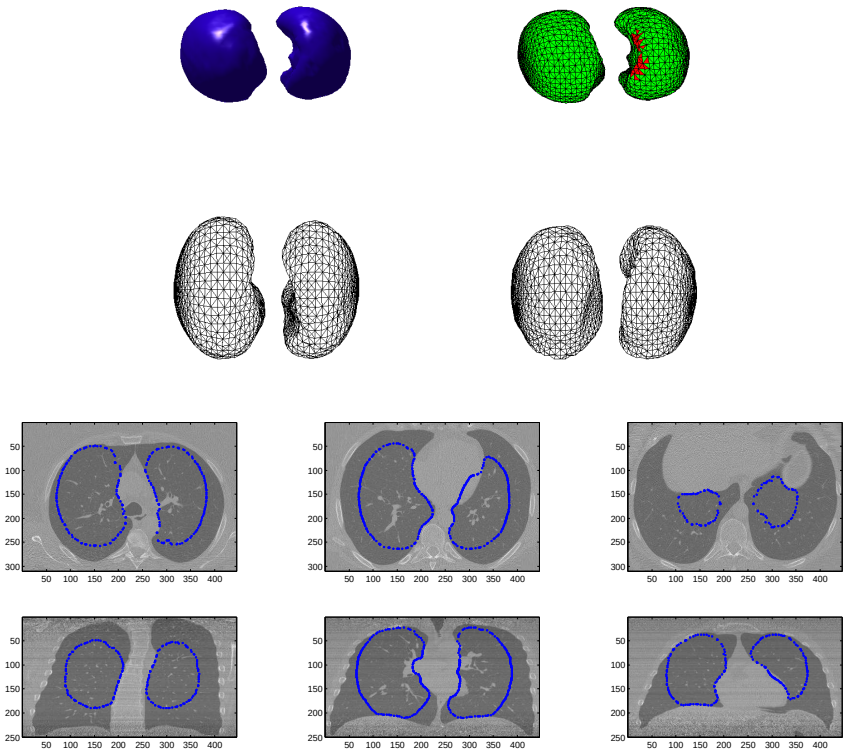


Figure 5.25: Lung segmentation: Surface at different viewing angles (row 1-2) and cross-sections (row 3: $z = 80, 150, 200$, row 4: $y = 80, 150, 200$) at $m = 100$ at time $t = 10$. The original images are from the Lung Image Database Consortium image collection (LIDC-IDRI) of The Cancer Imaging Archive (TCIA), see Reeves et al. (2007); Armato et al. (2011); Reeves and Biancardi (2011).

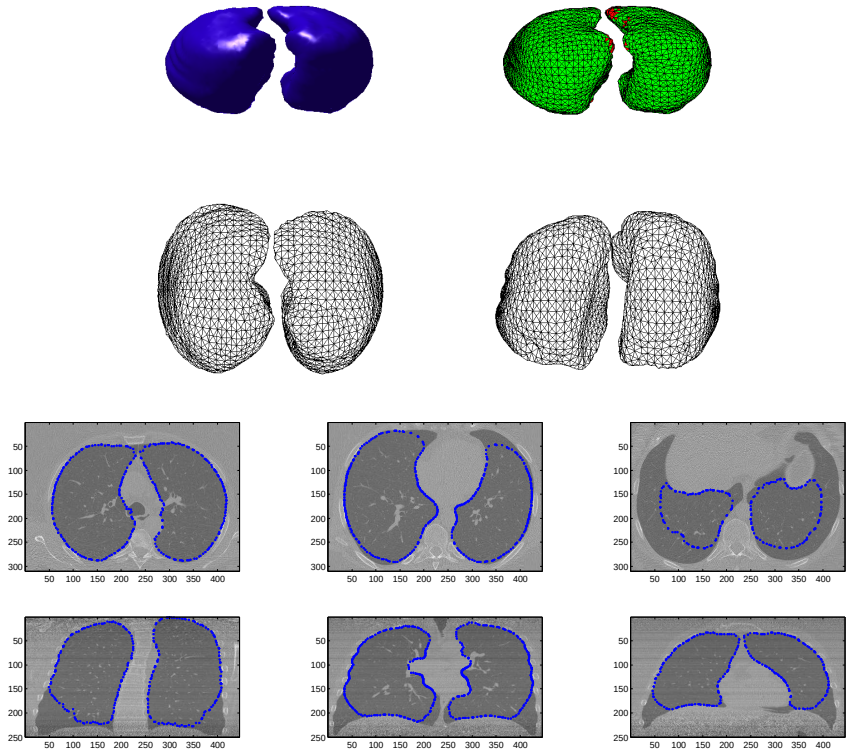


Figure 5.26: Lung segmentation: Surface at different viewing angles (row 1-2) and cross-sections (row 3: $z = 80, 150, 200$, row 4: $y = 80, 150, 200$) at $m = 300$ at time $t = 30$. The original images are from the Lung Image Database Consortium image collection (LIDC-IDRI) of The Cancer Imaging Archive (TCIA), see Reeves et al. (2007); Armato et al. (2011); Reeves and Biancardi (2011).

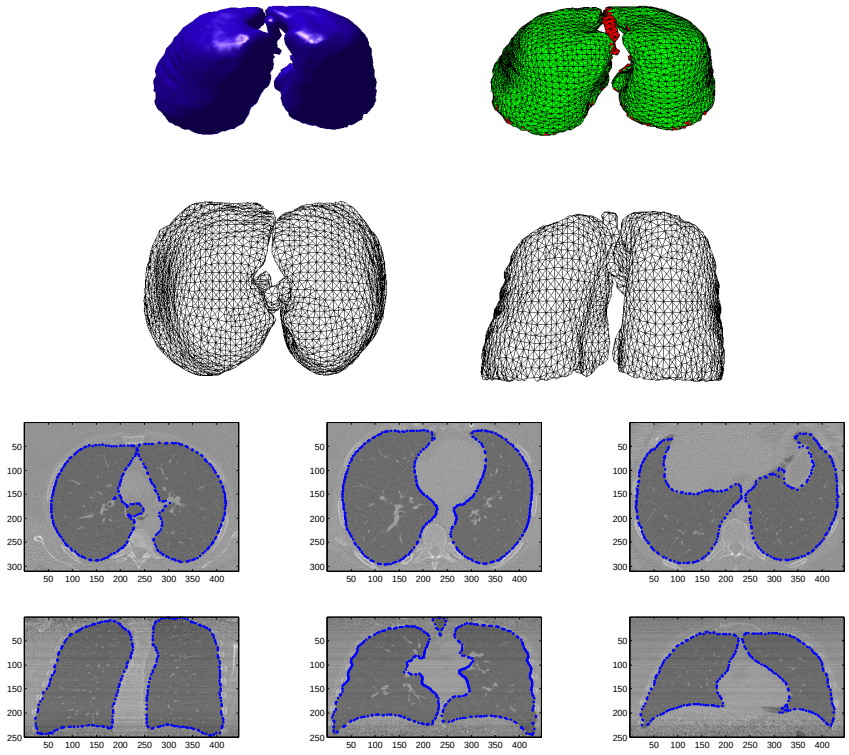


Figure 5.27: Lung segmentation: Surface at different viewing angles (row 1-2) and cross-sections (row 3: $z = 80, 150, 200$, row 4: $y = 80, 150, 200$) at $m = 600$ at time $t = 60$. The original images are from the Lung Image Database Consortium image collection (LIDC-IDRI) of The Cancer Imaging Archive (TCIA), see Reeves et al. (2007); Armato et al. (2011); Reeves and Biancardi (2011).

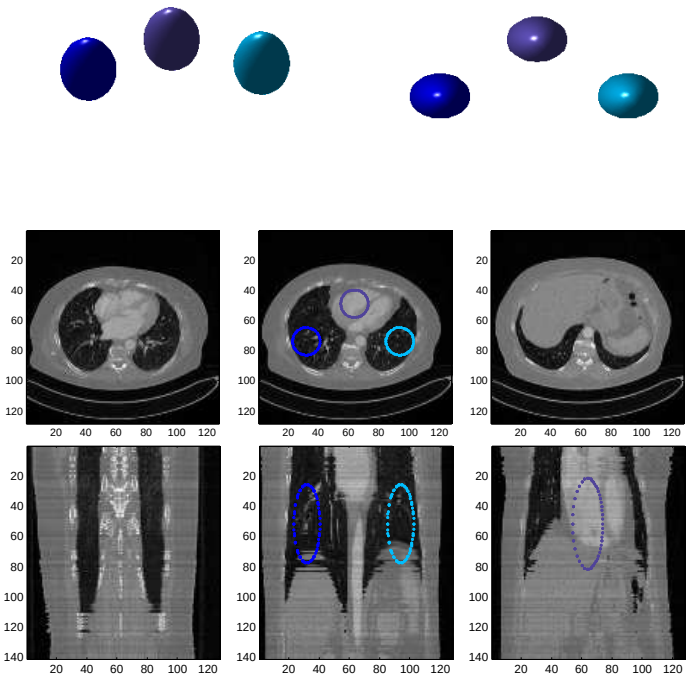


Figure 5.28: Lung and heart segmentation: Surface (row 1) and cross-sections (row 2: $z = 10, 40, 90$, row 3: $y = 38, 60, 80$) at $m = 0$ at time $t = 0$. Credits (original CT images): C. Stroszczynski, Radiology, University Hospital Regensburg.

Example using data from University Hospital Regensburg - Lung and heart segmentation

In the next examples, we consider CT images provided by C. Stroszczynski, Radiology, University Hospital Regensburg. The original data set consists of DICOM images, which are first transformed to 3D volumetric images.

Figure 5.28 - Figure 5.32 show the result and intermediate steps of a segmentation applied on a set of 141 CT images. The surfaces are differently colored to visualize the multiphase image segmentation. The 3D image consists of $128 \times 128 \times 141$ voxels. For the segmentation, we use $\sigma = 1$ and $\lambda = 30$ to weight the curvature term and the external forcing term and use an initial time step size of $\Delta t = 0.2$ with a time step control as described in Section 5.3.4. Therefore we set $\delta X_n^{\max} = 4$ and $\delta X_n^{\min} = 0.1$. Further, we perform mesh regularization with parameters $\alpha_0 = 0.01$ and $\phi_{\max} = 120°$. Figure 5.28 - 5.32 show the surfaces at time step $m = 0, 25, 50, 100, 200$ from different viewing angles and cross-sections of the image and of the surface. For the cross-sections, we consider the planes given by $y = 38, 60, 80$ and $z = 10, 40, 90$.

As can be seen in the subfigures showing the y-cross-sections, some slice images containing the top of the lung, are missing in the given CT data. This is why the final segmented surfaces appear flattened, cf. Figure 5.32.

It has to be noted, that we do not know the slice thickness of the CT slice images nor the size of one pixel of a slice image in the real world. We therefore handle the voxels of the 3D images as cubes of side length 1. Consequently, the x-, y- and z-directions are not scaled as in reality. The images showing y-cross sections appear stretched in z-direction.

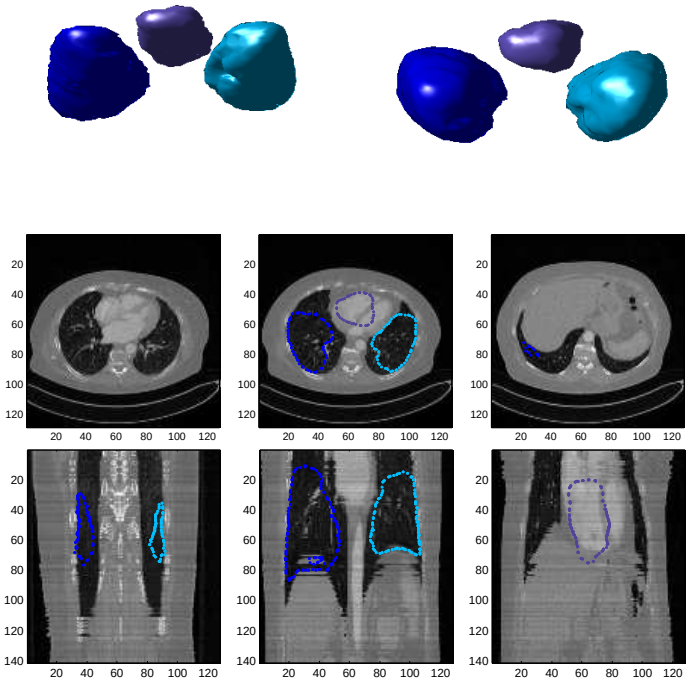


Figure 5.29: Lung and heart segmentation: Surface (row 1) and cross-sections (row 2: $z = 10, 40, 90$, row 3: $y = 38, 60, 80$) at $m = 25$ at time $t = 5$. Credits (original CT images): C. Stroszczynski, Radiology, University Hospital Regensburg.

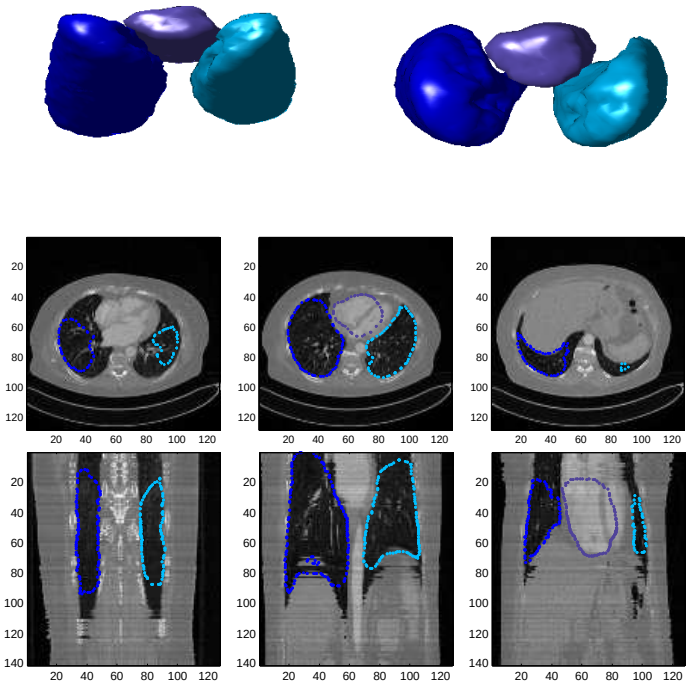


Figure 5.30: Lung and heart segmentation: Surface (row 1) and cross-sections (row 2: $z = 10, 40, 90$, row 3: $y = 38, 60, 80$) at $m = 50$ at time $t = 10$. Credits (original CT images): C. Stroszczynski, Radiology, University Hospital Regensburg.

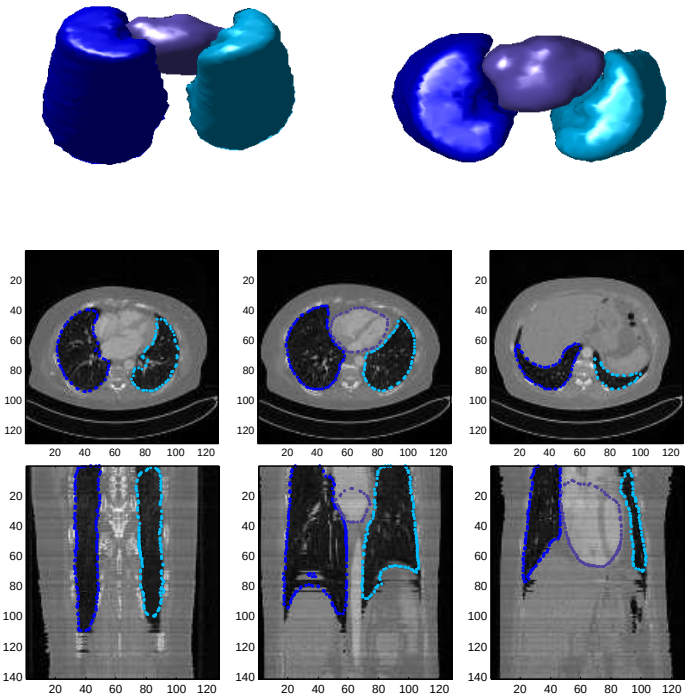


Figure 5.31: Lung and heart segmentation: Surface (row 1) and cross-sections (row 2: $z = 10, 40, 90$, row 3: $y = 38, 60, 80$) at $m = 100$ at time $t = 20$. Credits (original CT images): C. Stroszczynski, Radiology, University Hospital Regensburg.

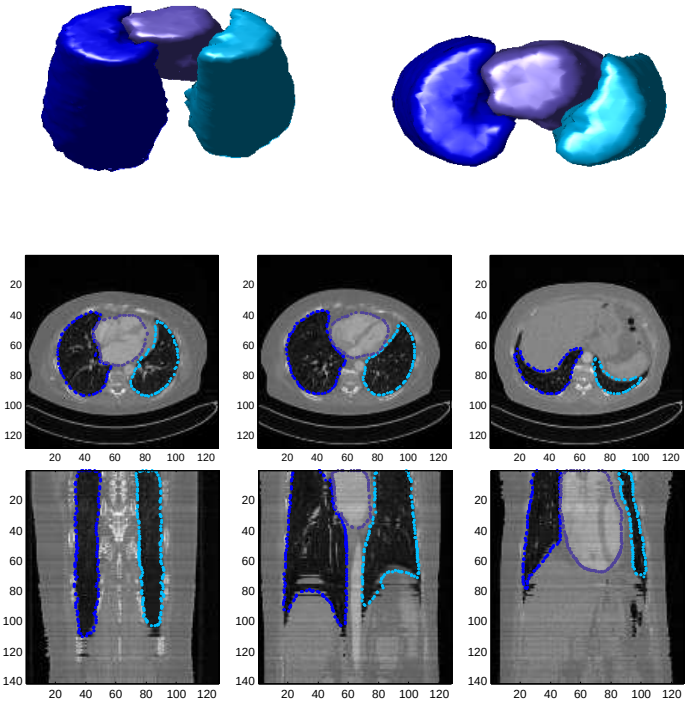


Figure 5.32: Lung and heart segmentation: Surface (row 1) and cross-sections (row 2: $z = 10, 40, 90$, row 3: $y = 38, 60, 80$) at $m = 200$ at time $t = 40$. Credits (original CT images): C. Stroszczynski, Radiology, University Hospital Regensburg.

Example using data from University Hospital Regensburg - Processing with Preprocessing

Considering the y-cross sections in Figures 5.28 - Figure 5.32, one can observe that the slice images are of different brightness resulting in horizontal stripe-like noise. In the example shown by Figures 5.28-5.32 the segmentation still works without any preprocessing of the images. But in general, such stripe-like noise can cause problems, in contrast to e.g. moderate Gaussian noise.

In those cases, where the brightness difference is higher, problems like false segmentation can occur. Figure 5.33 (left) shows a y-cross-section of another 3D image generated by computed tomography. Since our external forcing term in the evolution equation is based on the Chan-Vese segmentation, the mean gray value determines the connected regions in the image. Such bright stripes as in Figure 5.33 (left) could therefore lead to false image regions.

To handle this problem we can preprocess the image by considering the gray value of a representative set of a few pixels. In detail, we consider one reference image, here for example at $z = 200$. For this image, we compute the mean gray value, denoted by u_{ref}, in a small two-dimensional subset. Here we compute it in $[2, 8] \times [67, 73] \times \{200\}$, i.e. we choose a set close to the left image boundary. For each image $z = 1, \ldots, N_z$, where N_z is the number of slice images, let u_{loc} be the mean gray value in $[2, 8] \times [67, 73] \times \{z\}$. We multiply the slice image with $u_{\text{ref}}/u_{\text{loc}}$. Figure 5.33 (right) shows a y-cross section of the resulting preprocessed image.

Such a representative set like $[2, 8] \times [67, 73] \times \{200\}$ need not be large. Here we choose a rectangular set of 7×7 pixels. A single pixel or a set of only 2×2 pixels can be a too small set, since medical images are typically also affected by additional random noise. The representative set should be chosen not too small, since we want to compare the mean gray values u_{ref} and u_{loc}. With a set of 7×7 pixels, we obtained a satisfying result, see Figure 5.33 (right). Using the *mean* gray value, noise affecting single pixels is smoothed out.

Figure 5.34 shows the reference image at $z = 200$ and a brighter image at $z = 220$ as well as the resulting preprocessed image at $z = 220$.

Using now the preprocessed 3D image, we perform again an image segmentation. The resolution of this 3D image is $128 \times 128 \times 400$ voxels. We use the following parameters: weighting parameters $\sigma = 1$, $\lambda = 50$, time step size $\Delta t = 1$ with time step control, $\delta X_n^{\max} = 5$, $\delta X_n^{\min} = 0.5$. We perform mesh regularization with parameters $\alpha_0 = 0.1$ and $\phi_{\max} = 120°$. Figure 5.35 - 5.39 show the surfaces at time step $m = 0, 10, 50, 200, 500$ from different viewing angles and cross-sections of the image and of the surface. For the cross-sections, we consider the planes given by $y = 37, 44, 64$ and $z = 100, 200, 250$.

Example using data from University Hospital Regensburg - Segmentation of the abdominal region

In the next experiment, an abdominal region should be segmented. The resolution of the 3D image, that we now consider, is $128 \times 128 \times 80$ voxels. For the segmentation, we use the following parameters: As weighting parameters, we set $\sigma = 1$, $\lambda = 1000$. We use a time step size of $\Delta t = 1$ with time step control, $\delta X_n^{\max} = 5$, $\delta X_n^{\min} = 0.2$. No increase or decrease of the time step size is necessary during this experiment. As in the previous experiment, we use the parameters $\alpha_0 = 0.1$ and $\phi_{\max} = 120°$ for the mesh regularization. Figure 5.40 - 5.43 show the surfaces at time step $m = 0, 10, 50, 100$ from different viewing angles and

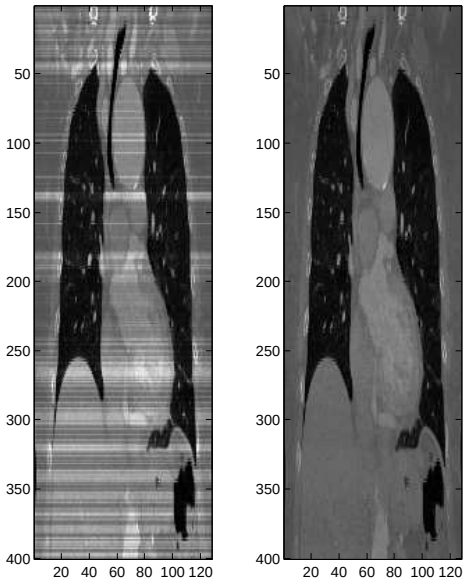


Figure 5.33: Original image (left) and preprocessed image with removed stripe-like noise (right). Cross-section $y = 64$. Credits (original CT image): C. Stroszczynski, Radiology, University Hospital Regensburg.

cross-sections of the image and of the surface. For the cross-sections, we draw the planes given by $y = 52, 64, 76$ and $z = 30, 50, 70$.

Example using data from University Hospital Regensburg - Splitting demonstration during lung segmentation

Finally, we consider an experiment where a topology change occurs. We perform again a lung segmentation starting with one initial surface which is split into two surfaces. Figure 5.44 - 5.51 show the surface(s) at time step $m = 0, 49, 50, 60, 100, 500, 900$ as well as cross-sections of the image and of the surface(s). For the cross-sections, we consider the planes given by $y = 50, 64, 80$ and $z = 80, 120, 160$.

The splitting occurs at time step $m = 50$. To detect the topology change, we use an auxiliary background grid with grid size $a = 2$. A cube of the grid is considered for possible topology changes if more than $N_{\text{detect}} = 8$ nodes are located inside the cube. Further, we use the parameters $thr1 = 30°$, $thr2 = 150°$ and $thr3 = 40°$, recall Section 5.3.3. Mesh regularization is suppressed at those parts where the normal vector changes rapidly, see Figure 5.47. After the splitting, the two surfaces grow and new triangles are created by bisection of too large triangles. For the segmentation we use the parameters $\sigma = 1$ and $\lambda = 20$. The time step size is set to $\Delta t = 0.2$ with time step control using $\delta X_n^{\max} = 2$, $\delta X_n^{\min} = 0.1$. However, no increase or decrease of the time step size is necessary. For mesh regularization $\alpha_0 = 0.01$ and $\phi_{\max} = 120°$ are applied.

As in the two-dimensional case, the choice of σ and λ strongly influences the smoothness of the surfaces and grade of details of the segmented objects' boundaries. By decreasing σ, the area term would be weighted less and the surfaces could continue to grow.

As postprocessing step, we compute the volume of the two enclosed regions and the area of the region boundaries. The right lung of the patient, i.e. the left surface in the Figure 5.51, has an area of $A_1 = 3.309 \cdot 10^4$ and a volume of $V_1 = 2.691 \cdot 10^5$. Note, that CT images

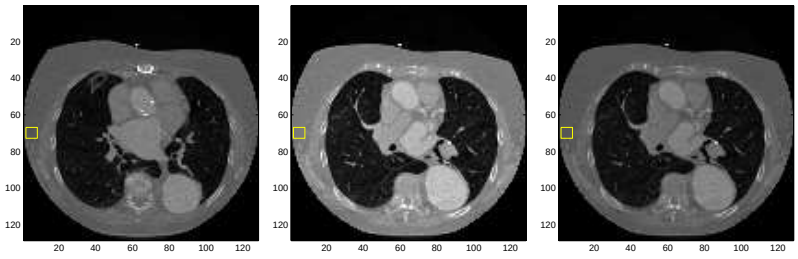


Figure 5.34: Illustration of the preprocessing. 1st subfigure: reference slice image at $z = 200$ with mean gray value of $u_{\text{ref}} = 0.2249$ in the yellow marked box. 2nd subfigure: image at $z = 220$ with mean gray value of $u_{\text{loc}} = 0.3232$ in the yellow marked box. 3rd subfigure: preprocessed image at $z = 220$ after multiplying the original 2D slice image with $u_{\text{ref}}/u_{\text{loc}} = 0.6959$. Credits (original CT images): C. Stroszczynski, Radiology, University Hospital Regensburg.

are mirror images. The left lung of the patient (right surface in the figure) has an area of $A_2 = 2.801 \cdot 10^4$ and a volume of $V_2 = 1.923 \cdot 10^5$. Thus, as expected, the volume of the right lung is larger compared to the left lung. Note, that we handle a voxel as a cube with side length 1, resulting in values of magnitude 10^4 for the area and 10^5 for the volume. If details on the acquisition system of the CT images are known (like the slice thickness, and the height and width of one pixel of a slice image), the area and the volume can be computed precisely and can be expressed in the metric system for practical interpretation of the values.

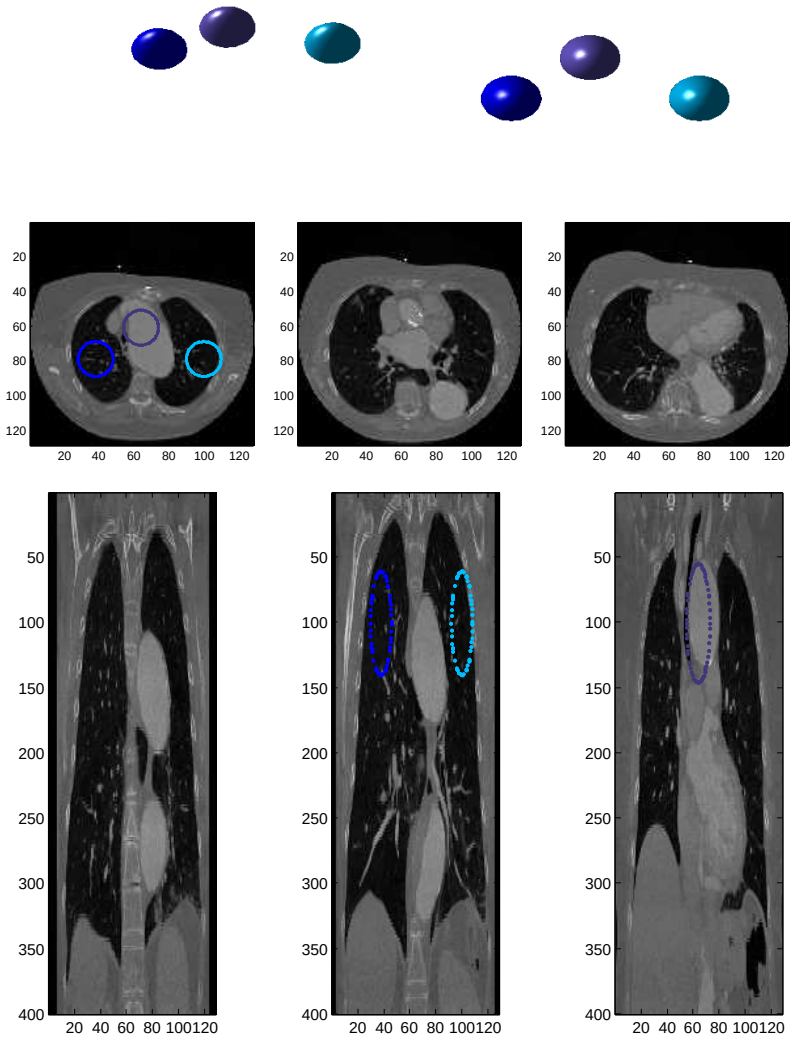


Figure 5.35: Medical image segmentation: Surface at different viewing angles (row 1) and cross-sections (row 2: $z = 100, 200, 250$, row 3: $y = 37, 44, 64$) at $m = 0$ at time $t = 0$. Credits (original CT images): C. Stroszczynski, Radiology, University Hospital Regensburg.

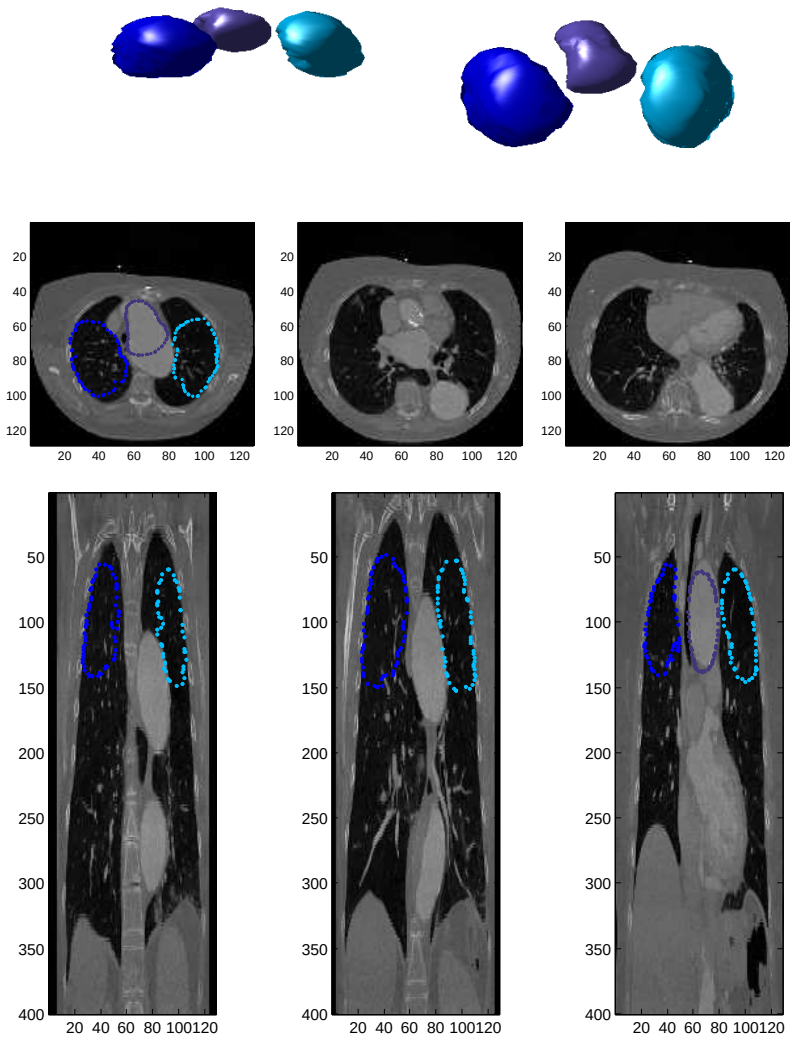


Figure 5.36: Medical image segmentation: Surface at different viewing angles (row 1) and cross-sections (row 2: $z = 100, 200, 250$, row 3: $y = 37, 44, 64$) at $m = 10$ at time $t = 10$. Credits (original CT images): C. Stroszczynski, Radiology, University Hospital Regensburg.

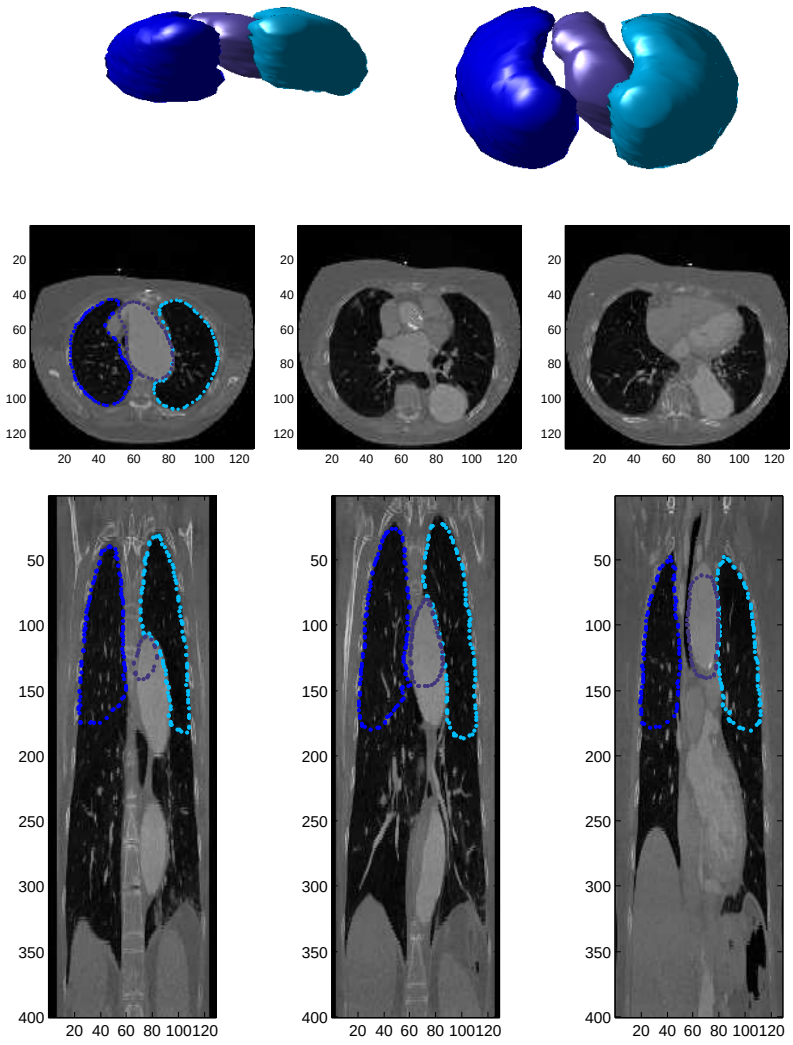


Figure 5.37: Medical image segmentation: Surface at different viewing angles (row 1) and cross-sections (row 2: $z = 100, 200, 250$, row 3: $y = 37, 44, 64$) at $m = 50$ at time $t = 50$. Credits (original CT images): C. Stroszczynski, Radiology, University Hospital Regensburg.

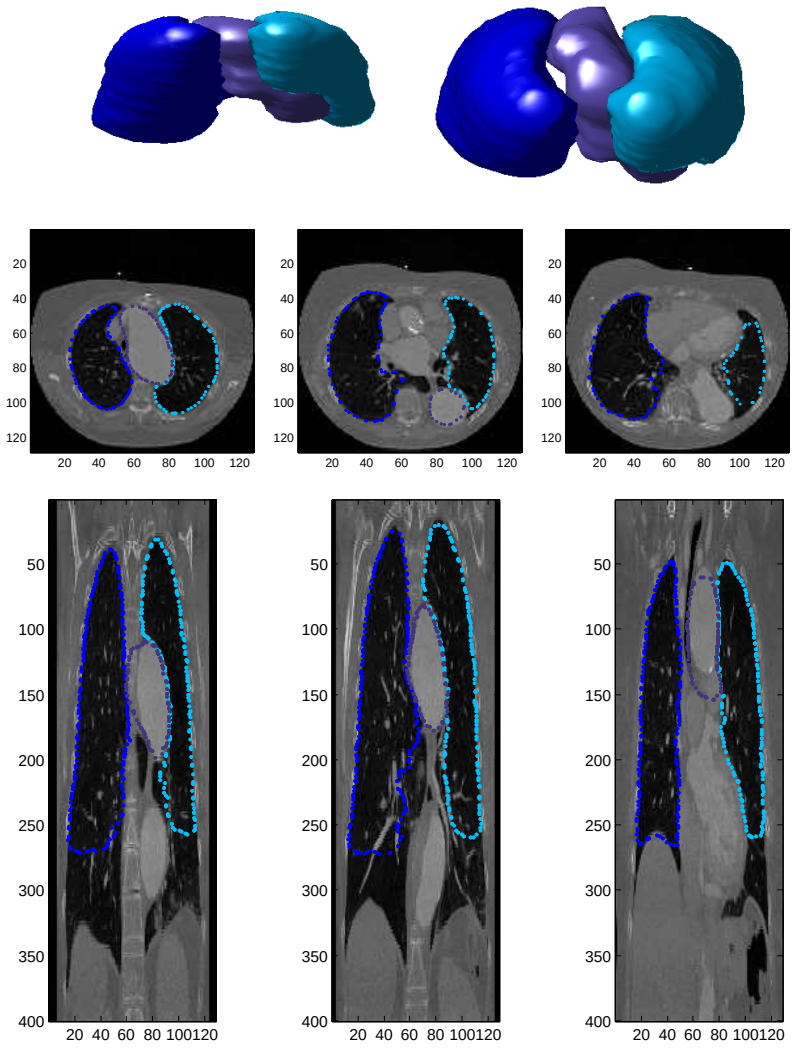


Figure 5.38: Medical image segmentation: Surface at different viewing angles (row 1) and cross-sections (row 2: $z = 100, 200, 250$, row 3: $y = 37, 44, 64$) at $m = 200$ at time $t = 200$. Credits (original CT images): C. Stroszczynski, Radiology, University Hospital Regensburg.

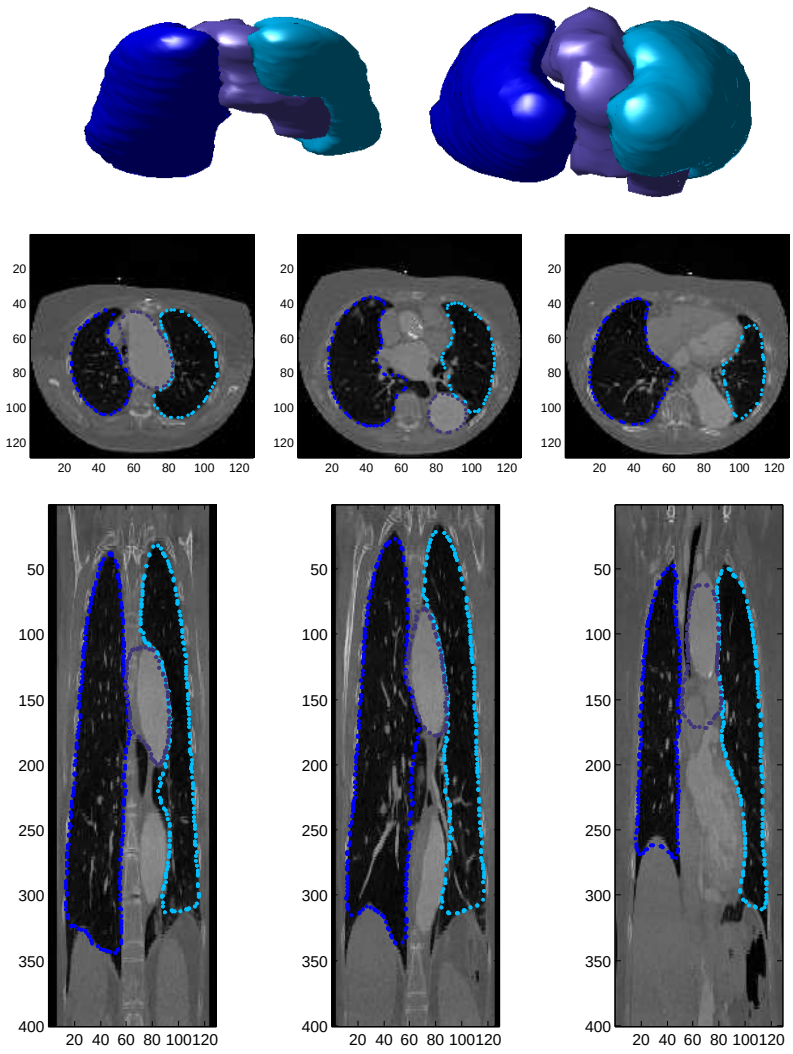


Figure 5.39: Medical image segmentation: Surface at different viewing angles (row 1) and cross-sections (row 2: $z = 100, 200, 250$, row 3: $y = 37, 44, 64$) at $m = 500$ at time $t = 500$. Credits (original CT images): C. Stroszczynski, Radiology, University Hospital Regensburg.

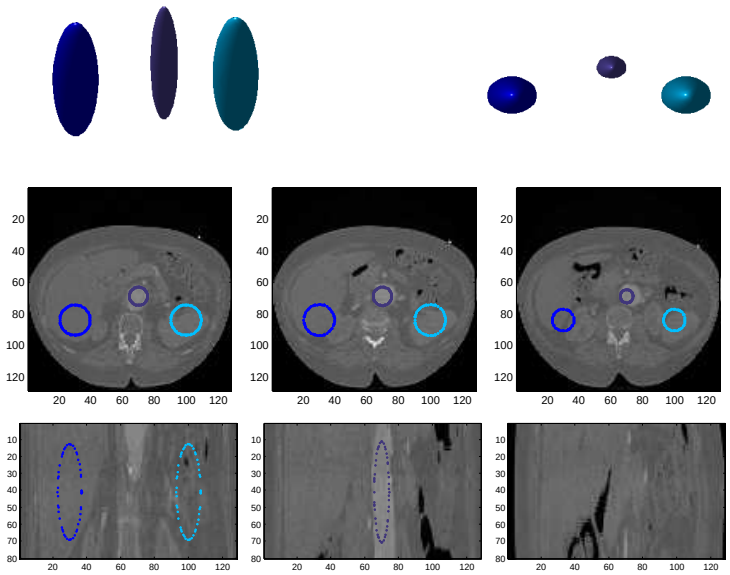


Figure 5.40: Medical image segmentation (abdominal region): Surface at different viewing angles (row 1) and cross-sections (row 2: $z = 30, 50, 70$, row 3: $y = 52, 64, 76$) at $m = 0$ at time $t = 0$. Credits (original CT images): C. Stroszczynski, Radiology, University Hospital Regensburg.

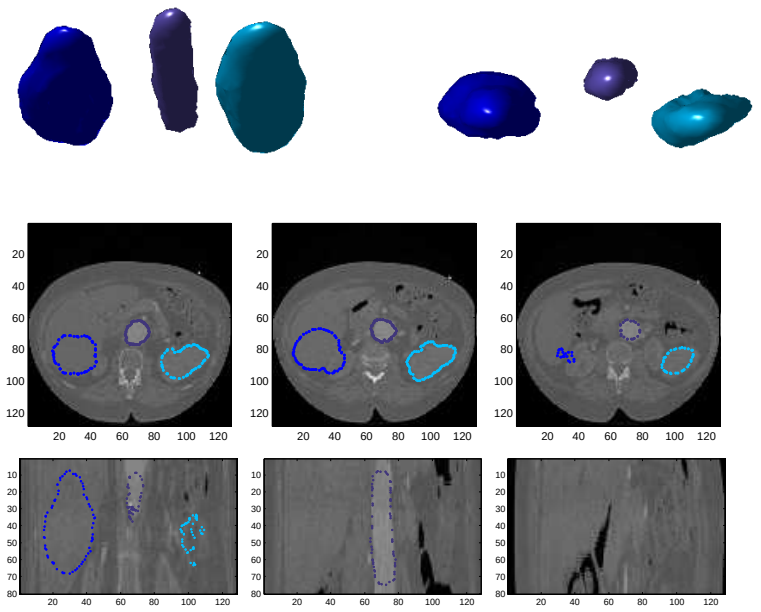


Figure 5.41: Medical image segmentation (abdominal region): Surface at different viewing angles (row 1) and cross-sections (row 2: $z = 30, 50, 70$, row 3: $y = 52, 64, 76$) at $m = 10$ at time $t = 10$. Credits (original CT images): C. Stroszczynski, Radiology, University Hospital Regensburg.

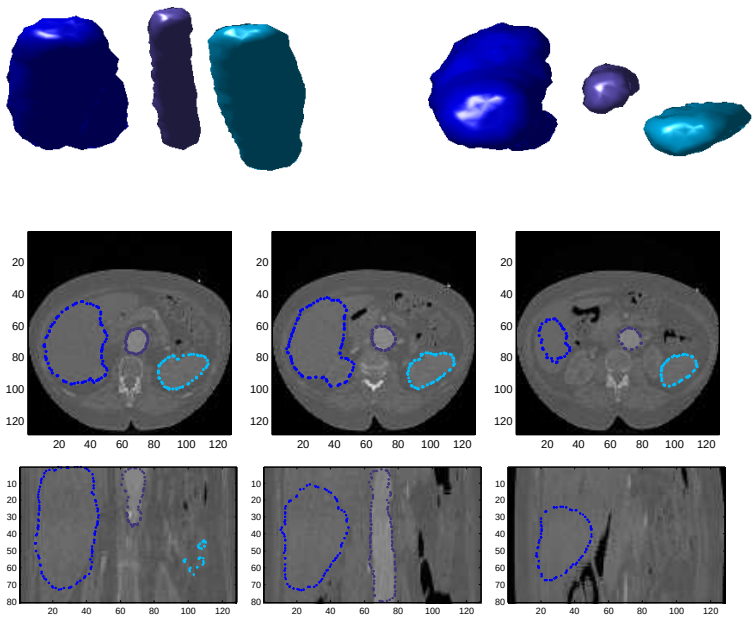


Figure 5.42: Medical image segmentation (abdominal region): Surface at different viewing angles (row 1) and cross-sections (row 2: $z = 30, 50, 70$, row 3: $y = 52, 64, 76$) at $m = 50$ at time $t = 50$. Credits (original CT images): C. Stroszczynski, Radiology, University Hospital Regensburg.

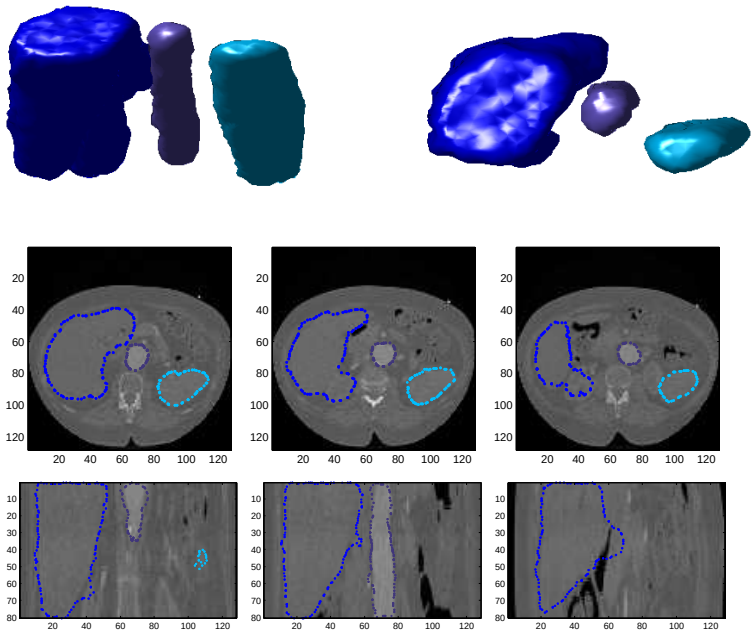


Figure 5.43: Medical image segmentation (abdominal region): Surface at different viewing angles (row 1) and cross-sections (row 2: $z = 30, 50, 70$, row 3: $y = 52, 64, 76$) at $m = 100$ at time $t = 100$. Credits (original CT images): C. Stroszczynski, Radiology, University Hospital Regensburg.

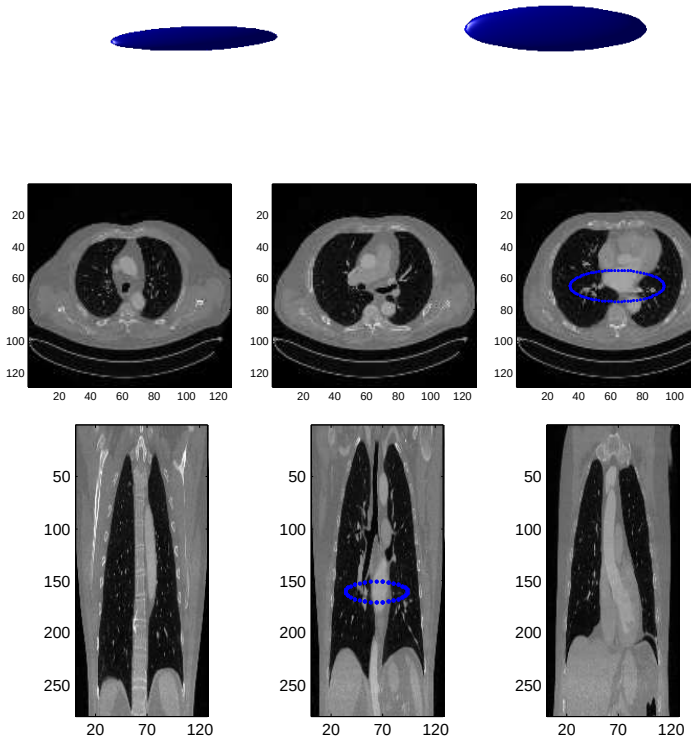


Figure 5.44: Lung segmentation with splitting: Surface at different viewing angles (row 1) and cross-sections (row 2: $z = 80, 120, 160$, row 3: $y = 50, 64, 80$) at $m = 0$ at time $t = 0$. Credits (original CT images): C. Stroszczynski, Radiology, University Hospital Regensburg.

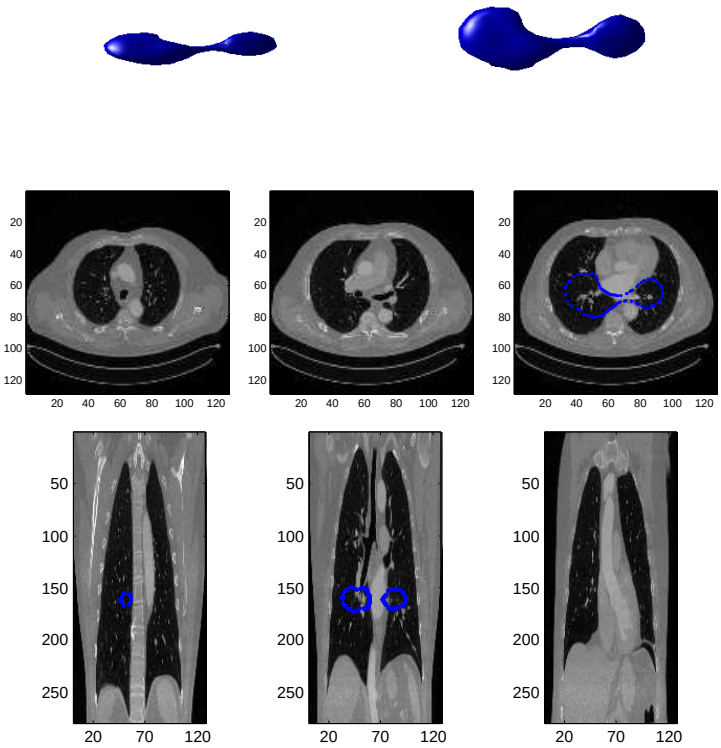


Figure 5.45: Lung segmentation with splitting: Surface at different viewing angles (row 1) and cross-sections (row 2: $z = 80, 120, 160$, row 3: $y = 50, 64, 80$) at $m = 49$ at time $t = 9.8$. Credits (original CT images): C. Stroszczynski, Radiology, University Hospital Regensburg.

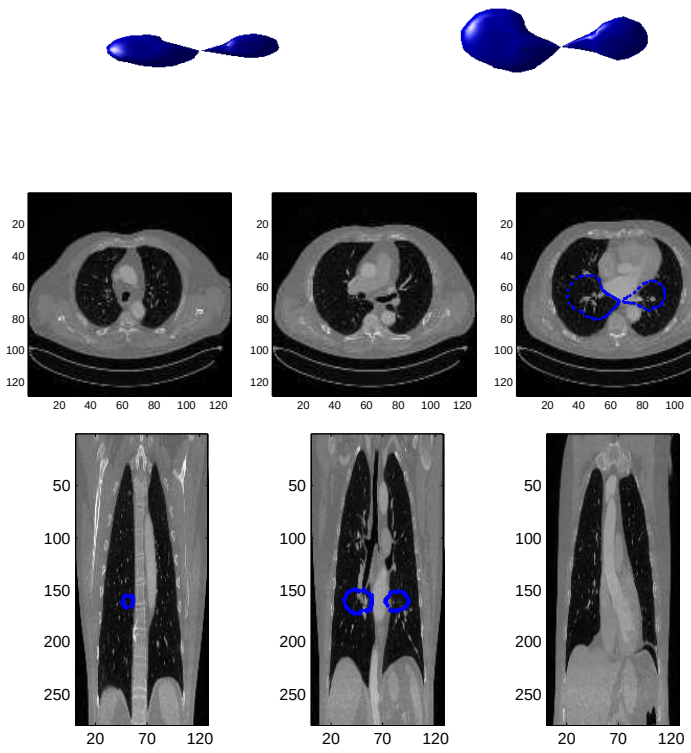


Figure 5.46: Lung segmentation with splitting: Surface at different viewing angles (row 1) and cross-sections (row 2: $z = 80, 120, 160$, row 3: $y = 50, 64, 80$) at $m = 50$ at time $t = 10$. Credits (original CT images): C. Stroszczynski, Radiology, University Hospital Regensburg.

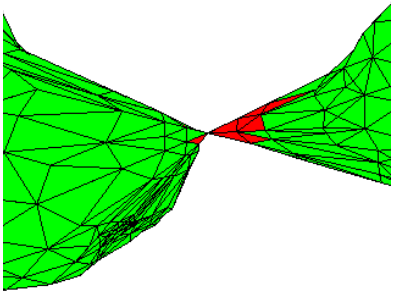


Figure 5.47: Lung segmentation with splitting: Situation at $m = 50$ immediately after a splitting has been performed. Magnification close to the splitting location. Red marked parts: the induced tangential motion is suppressed. Green marked parts: Tangential motion of the vertices is allowed.

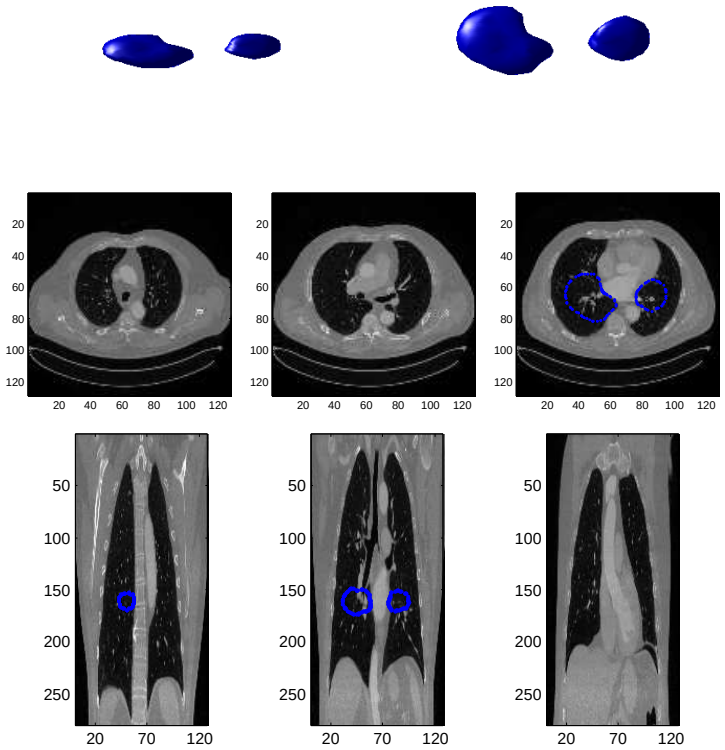


Figure 5.48: Lung segmentation with splitting: Surface at different viewing angles (row 1) and cross-sections (row 2: $z = 80, 120, 160$, row 3: $y = 50, 64, 80$) at $m = 60$ at time $t = 12$. Credits (original CT images): C. Stroszczynski, Radiology, University Hospital Regensburg.

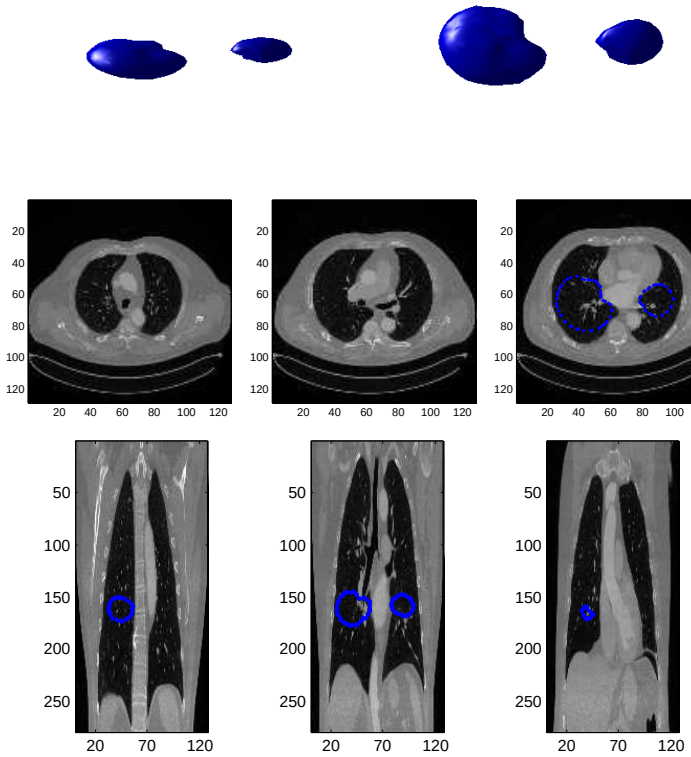


Figure 5.49: Lung segmentation with splitting: Surface at different viewing angles (row 1) and cross-sections (row 2: $z = 80, 120, 160$, row 3: $y = 50, 64, 80$) at $m = 100$ at time $t = 20$. Credits (original CT images): C. Stroszczynski, Radiology, University Hospital Regensburg.

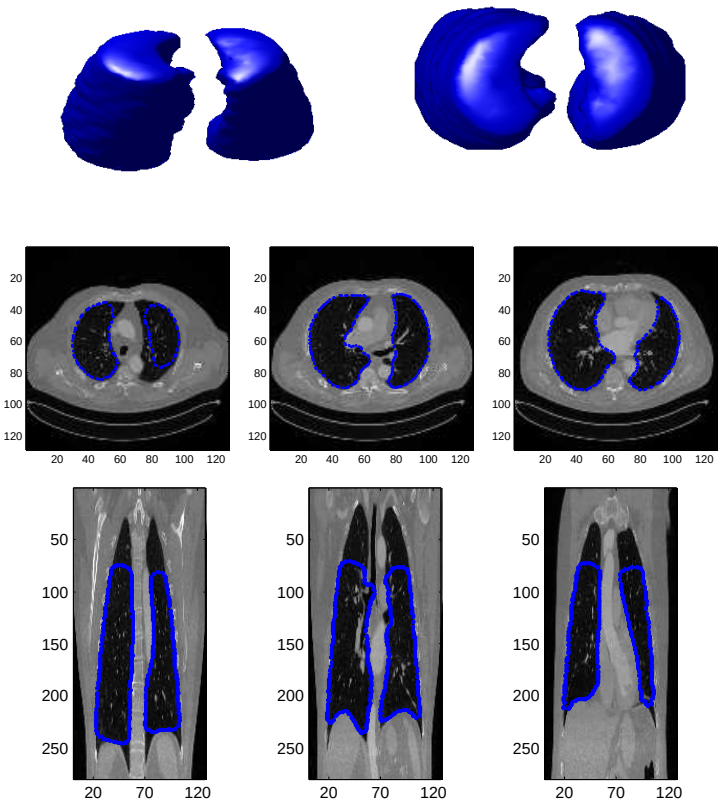


Figure 5.50: Lung segmentation with splitting: Surface at different viewing angles (row 1) and cross-sections (row 2: $z = 80, 120, 160$, row 3: $y = 50, 64, 80$) at $m = 500$ at time $t = 100$. Credits (original CT images): C. Stroszczynski, Radiology, University Hospital Regensburg.

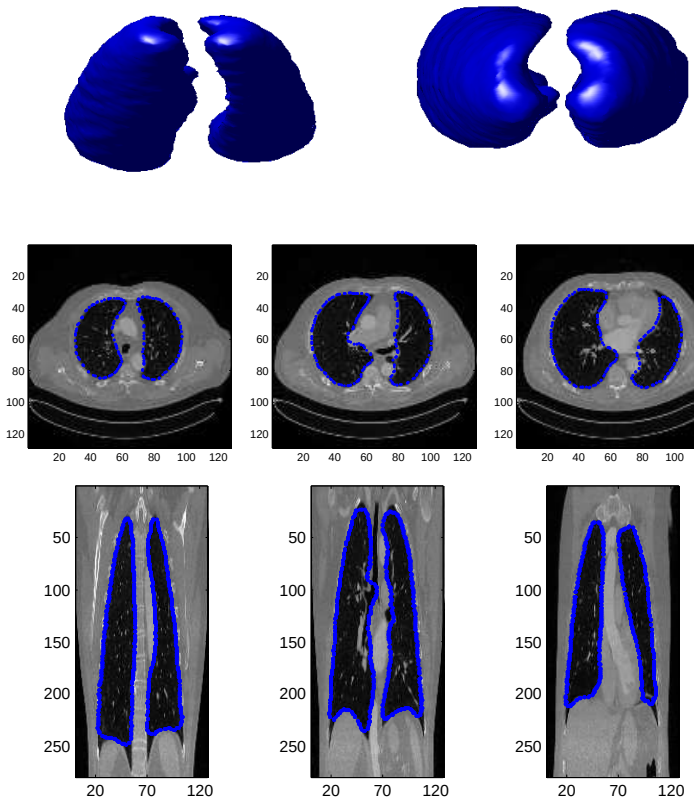


Figure 5.51: Lung segmentation with splitting: Surface at different viewing angles (row 1) and cross-sections (row 2: $z = 80, 120, 160$, row 3: $y = 50, 64, 80$) at $m = 900$ at time $t = 180$. Credits (original CT images): C. Stroszczynski, Radiology, University Hospital Regensburg.

Chapter 6

Conclusion and Outlook

In this thesis, parametric methods for image segmentation with a subsequent image restoration were developed. The thesis covers two-dimensional planar images, images defined on surfaces and three-dimensional images. Many concepts and methods developed first for the two-dimensional case could be used and extended to the non-Euclidean case (images on surfaces) and to the higher dimensional case (three-dimensional images).

The main advantage of parametric methods is that the resulting schemes are very efficient from a computational point of view: the schemes have a small computational time and low memory requirements, since the evolution of curves is only a one-dimensional problem and the evolution of surfaces is only a two-dimensional problem. Further, one can easily incorporate triple junctions and curves with free endpoints into the curve evolution scheme. Also the extension to multiphase image segmentation can be performed easily and efficiently. The computational effort is mainly dependent on number of mesh points to discretize the hypersurfaces. The effort is not dependent on the number of different regions. Therefore, a multiphase segmentation does not result in a substantially greater effort compared to a two-phase segmentation. The mentioned properties of parametric methods were compared to indirect approaches for evolution of curves and surfaces like the level set method. Using standard level set methods, the handling of junctions, free endpoints and multiple phases often requires more than one level set function and even the use of artificial regions and artificial curves in the case of free endpoints. Additionally, parametric methods directly represent the hypersurfaces and allow for a direct postprocessing of the segmentation result. For example, the size of the boundaries and the area/volume of the segmented regions can be determined fast and easily in a postprocessing step.

We further extended the method of Mikula and Urbán (2012) for detection of topology changes to handle different types of such changes: Apart from splitting and merging of curves in $\mathbb{R}^2$, we can also handle the emergence of triple junctions and intersections with the image boundary as well as deletion of curves in complex curve networks. We used the idea of an artificial background grid and developed an efficient method to handle also topology changes of curves on surfaces and topology changes of surfaces in $\mathbb{R}^3$. In case of surfaces in $\mathbb{R}^3$, we considered splitting, merging, decrease and increase of the genus of a surface (for example: sphere to torus).

Another feature of our method is the good mesh quality of the curves and the surfaces. For evolving curves in the plane and for curves on surfaces, we could show equidistribution of the mesh points along the discretized curves. For triangulated surfaces, problems concerning the mesh quality only appeared in special situations, for example, when a pinch-off occurs,

shortly before a splitting of a surface. We proposed an efficient method for mesh regularization via an induced tangential motion of the mesh points. The tangential flow of the nodes was incorporated in the discrete scheme such that no additional routine has to be executed.

In a variety of experiments, we presented how different gray-scaled and colored images can be segmented. We presented how the different topology changes can be successfully detected and executed. We applied our method for image segmentation with a subsequent image denoising on a number of artificial images to study the general behavior of the algorithm, and applied it on a number of real images to show that the developed approach can be used also for practical problems. We considered different types of images from navigation, Earth observation, medicine and other areas.

A possible extension of the 3D algorithm could be the support of so-called double or triple bubbles. This is the generalization of triple junctions to a higher dimension. A possible way to extend the scheme to handle multi-bubbles is described in Barrett et al. (2010b).

Some further work and improvements could be invested in the programming. The experiments presented in this thesis were performed using a MATLAB implementation of the algorithms. The computations could be accelerated using C/C++, since for example many *for*-loops are used. Such loops are needed for example, when the contribution of each mesh point to the system matrix of the linear system should be computed. Also an outer loop is needed to realize the time discretization. Such loops slow down a MATLAB program. MATLAB has been chosen for demonstration and visualization reasons. For applications which require a very fast execution of the software, like real-time applications, a C/C++ implementation will be likely required and in most cases some modifications for the real-time PC target with a real-time operational system need to be performed. Further, parallel computing or hardware computing (for example usage of field-programmable gate arrays (FPGAs)) for image processing pose a wide area of additional research. However, these areas are out of the scope of this thesis.

Image processing is a very interdisciplinary research area with many challenges concerning mathematics (modeling, optimization problems, differential equations, numerical issues), computer science (implementation, software/hardware adaptations and improvements) and engineering (work on sensors and devices for image capturing and image generation). Here, we mainly considered mathematical issues which arise from the evolution of curves and surfaces.

Many contributions of this thesis can also be applied for other problems involving hypersurfaces. The developed methods are not restricted to image processing problems only. For example, the developed methods for handling of topology changes can be used for any other application where the standard parametric method develops singularities like a pinch-off. Therefore, we also presented and studied an example from mean curvature flow of surfaces, to demonstrate how one can proceed after a pinch-off occurs.

While diffusion problems in Euclidean spaces are meanwhile state of the art, only a few approaches can be found for solving diffusion problems on submanifolds like on surfaces in $\mathbb{R}^3$. The presented method for image denoising can be applied to smooth any kind of noisy data given on a surface. The data need not be a classical image.

In summary, we developed and analyzed efficient methods for image processing problems which can also be applied for other problems where evolution of curves or surfaces occur or where a smoothing of some data is required.

Symbols and Notation

General Notation

δ_{ij}	Kronecker delta, 1 if $i = j$ and 0 if $i \neq j$.
$\mathbb{R}^d$	d-dimensional Euclidean vector space.
$\vec{e}_i$, $i = 1, \ldots, d$	standard basis vectors in $\mathbb{R}^d$.
$\mathbb{R}^{d\times d}$	space of real $d \times d$-matrices.
$S^{d-1} \subset \mathbb{R}^d$	$(d-1)$-dimensional sphere in $\mathbb{R}^d$.
$\vec{x} \,.\, \vec{y} = \vec{x}^T \vec{y}$	Euclidean inner product of vectors $\vec{x}, \vec{y} \in \mathbb{R}^d$.
$\vec{x} \times \vec{y}$	cross product of vectors $\vec{x}, \vec{y} \in \mathbb{R}^d$.
$\|\vec{x}\|$	Euclidean norm of a vector $\vec{x} \in \mathbb{R}^d$.
$\vec{x}^\perp = (-x_2, x_1)^T$	counterclockwise rotation of a vector $\vec{x} = (x_1, x_2)^T \in \mathbb{R}^2$ by $\pi/2$.
$\vec{\mathrm{Id}}_d$	identity mapping of $\mathbb{R}^d$; can also be interpreted as identity matrix in $\mathbb{R}^{d\times d}$.
$\det(M)$	determinant of a matrix M.
$\overline{U}$	closure of a set $U \subset \mathbb{R}^d$.
∂U	topological boundary of U.
$U_0 \subset\subset U$	the closure $\overline{U_0}$ is a subset of the set U.
$U \setminus V$	complement of V in U for two sets U, V.
$V^\perp$	orthogonal complement of $V \subset \mathbb{R}^d$ with respect to the Euclidean inner product.
Ω	subset of $\mathbb{R}^d$.
I	interval in $\mathbb{R}$ or other one-dimensional reference manifold like $\mathbb{R}/\mathbb{Z}$.
Γ	$(d-1)$-dimensional hypersurface or union of hypersurfaces.
$\|\Gamma\|$	area of the hypersurfaces belonging to Γ, see Definition 2.8; in case of curves: length of the curves.
$T_{\vec{p}}\Gamma$	tangential space of Γ at $\vec{p}$, see Definition 2.4.
$N_{\vec{p}}\Gamma = (T_{\vec{p}}\Gamma)^\perp$	normal vector space of Γ at $\vec{p}$, see Definition 2.4.
$T_{\vec{p}}\partial\Gamma$	tangential space of $\partial\Gamma$ at $\vec{p} \in \partial\Gamma$.
$\vec{\nu}$	a normal vector field on the hypersurface Γ.

$\vec{\tau}$	a tangent vector field on the hypersurface Γ.
$\vec{\mu}$	outer co-normal in $T_{\vec{p}}\Gamma$, orthogonal to elements in $T_{\vec{p}}\partial\Gamma$, $\vec{p} \in \partial\Gamma$, see Remark 2.21.
$W^{1,2}(U)$	Sobolev space on U, $W^{k,p}(U)$ for $k = 1$ and $p = 2$.
$L^2(U)$	Lebesgue space on U, $L^p(U)$ for $p = 2$.
$L^\infty(U)$	space of all essentially bounded functions.
$SBV(U)$	space of special functions of bounded variation.
$C^k(U)$, $C^\infty(U)$	set of all k-times continuously differentiable functions, set of infinitely continuously differentiable functions on U.
$C^{1,1}(U)$	space of continuously differentiable functions which (partial) derivatives are Lipschitz-continuous. Hölder space $C^{k,\alpha}(U)$ for $k = 1$ and $\alpha = 1$.
$\mathrm{d}A$	area element, see Definition 2.7.
$\mathrm{d}s$	length element, see Remark 2.9.
$\nabla f = (\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_d})^T$	gradient of a function $f : \Omega \to \mathbb{R}$.
$\nabla_{\vec{x}} f$	gradient with respect to variable $\vec{x}$; notation used if f depends on different variables.
$\Delta f = \sum_{j=1}^{d} \frac{\partial^2 f}{\partial x_j^2}$	Laplace operator applied on $f : \Omega \to \mathbb{R}$.
$f_\rho = \frac{\partial f}{\partial \rho}$	(partial) derivative with respect to a real variable ρ.
$f_s = \frac{\partial f}{\partial s} = \frac{1}{\|\vec{x}_\rho\|} \frac{\partial f}{\partial \rho}$	derivative of a function $f : I \to \mathbb{R}$ with respect to arc-length s, where $\vec{x} : I \to \mathbb{R}^d$ is a paramerization of a curve Γ, see Remark 2.20.
$\partial_{\vec{\tau}} f(\vec{p})$	directional derivative, derivative of $f : \Gamma \to \mathbb{R}$ in direction $\vec{\tau} \in T_{\vec{p}}\Gamma$ at $\vec{p} \in \Gamma$, see Definition 2.10.
$\nabla_\Gamma f$	tangential gradient of $f : \Gamma \to \mathbb{R}$, also denoted by $(\underline{D}_1 f, \ldots, \underline{D}_d f)$, see Definition 2.11.
$\nabla_\Gamma \,.\, \vec{f}$	surface divergence of $\vec{f} : \Gamma \to \mathbb{R}^d$, see Definition 2.13.
$\Delta_\Gamma f$	Laplace-Beltrami operator of $f : \Gamma \to \mathbb{R}$, see Definition 2.14.
V_n	normal velocity of an evolving hypersurface, see Definition 2.24. Also defined for curves on surfaces, cf. Section 4.2.
χ_A	characteristic function for a subset $A \subset \mathbb{R}^d$.
$\mathrm{sign}(a)$	sign of a real number a.
$\mathrm{ceil}(a)$	smallest integer larger or equal to a.
$[0, T]$	time interval for time-dependent hypersurfaces.
t_m	$m = 1, \ldots, M$, discrete time points with $t_0 = 0$ and $t_M = T$.
τ_m, Δt	$= t_{m+1} - t_m$, $m = 0, \ldots, M-1$, time step size.
$c_k^{(m)}$	mean of scalar image function u_0 in the region $\Omega_k^{(m)}$.
$c_{k,j}^{(m)}$	mean of the j-th channel of the image function $\vec{u}_0$ in the region $\Omega_k^{(m)}$.
$b_k^{(m)}$	mean of the brightness component of the image function (using CB color space) in the region $\Omega_k^{(m)}$.
$\vec{v}_k^{(m)}$	mean of the chromaticity component of the image function (using CB color space) in the region $\Omega_k^{(m)}$.

$\vec{h}_k^{(m)}$	mean of the hue component of the image function (using HSV color space) in the region $\Omega_k^{(m)}$.
$s_k^{(m)}$	mean of the saturation component of the image function (using HSV color space) in the region $\Omega_k^{(m)}$.
$v_k^{(m)}$	mean of the value component of the image function (using HSV color space) in the region $\Omega_k^{(m)}$.
σ	weighting factor for the curvature term in the evolution equations.
λ	weighting factor for the external term in the evolution equations (in case of a scalar image function or in case of equal weighting factors of different external terms).
λ_C	weighting factor for the external term in the evolution equations (chromaticity part, CB space).
λ_B	weighting factor for the external term in the evolution equations (brightness part, CB space).
λ_H	weighting factor for the external term in the evolution equations (hue part, HSV space).
λ_S	weighting factor for the external term in the evolution equations (saturation part, HSV space).
λ_V	weighting factor for the external term in the evolution equations (value part, HSV space).
$F^{(m)}$, $F_i^{(m)}$	external term in the evolution equations, external term with respect to hypersurface $\Gamma_i^{(m)}$.
$a > 0$	grid size of auxiliary background grid (2D and 3D grids) used to detect topology changes.

Notation for planar curves in $\mathbb{R}^2$

$\Omega \subset \mathbb{R}^2$	two-dimensional image domain.
Γ	curve in Ω or union of curves.
$\mathrm{Int}(\Gamma)$	interior of an oriented curve Γ.
N_R	number of regions.
N_C	number of curves.
N_T	number of triple junctions.
N_I	number of boundary intersection points.
κ	mean curvature of Γ.
$\vec{\kappa} = \kappa\vec{\nu}$	mean curvature vector.
$\vec{x}$	parameterization of the curve (or set of curves) Γ.
I_i	one-dimensional reference manifold of a curve Γ_i, $i = 1, \ldots, N_C$, such that the curve can be parameterized by an $\vec{x}_i : I_i \to \mathbb{R}^2$.
$\vec{n}_{\partial\Omega}$	outer normal vector field on the boundary $\partial\Omega$ of the image domain.
$\vec{\nu}$	a normal vector field on the curve Γ.
$\vec{\tau}$	a tangent vector field on the curve Γ.
V_n	velocity of an evolving curve in direction $\vec{\nu}$ (normal velocity).
$V_{\tan}$	velocity of an evolving curve in direction $\vec{\tau}$ (tangential velocity).
$\vec{\Lambda}_k$	triple junctions, $k = 1, \ldots, N_T$.
$\vec{Q}_k$	boundary intersection points, $k = 1, \ldots, N_I$.
$P_n = \vec{\mathrm{Id}}_2 - \vec{x}_s \otimes \vec{x}_s$	projection operator, projects a vector or a vector-valued mapping onto the part normal to a curve Γ parameterized by $\vec{x}$.
$(\vec{\eta}, \vec{\chi})_{2,nor}$	$= \int_\Gamma \vec{P}_n\,\vec{\eta}\,.\,\vec{P}_n\,\vec{\chi}\,\mathrm{d}s$, an inner product for functions $\vec{\eta}$, $\vec{\chi} : I \to \mathbb{R}^2$.
$(\vec{\eta}, \vec{\chi})_{2,\Gamma,\partial\Gamma}$	$= \int_\Gamma \vec{\eta}\,.\,\vec{\chi}\,\mathrm{d}s + \vec{\eta}(1)\,.\,\vec{\chi}(1) + \vec{\eta}(0)\,.\,\vec{\chi}(0)$.
$\int_\Gamma u\,.\,v\,\mathrm{d}s$	$= \sum_{i=1}^{N_C} \int_{I_i} u_i\,.\,v_i\;\|(\vec{x}_i)_\rho\|\,\mathrm{d}\rho$ for scalar or vector-valued functions $u = (u_1, \ldots, u_{N_C}), v = (v_1, \ldots, v_{N_C}) \in L^2(I_1, \mathbb{R}^{(2)}) \times \ldots \times L^2(I_{N_C}, \mathbb{R}^{(2)})$.
$(\nabla_s u\,.\,\nabla_s v)_{\vert\Gamma_i}$	$(u_i)_s\,.\,(v_i)_s$ for $u = (u_1, \ldots, u_{N_C})$, $v = (v_1, \ldots, v_{N_C})$.
W	$= H^1(I_1, \mathbb{R}) \times \ldots \times H^1(I_{N_C}, \mathbb{R})$.
$\underline{V}$	certain subspace of $H^1(I_1, \mathbb{R}^2) \times \ldots \times H^1(I_{N_C}, \mathbb{R}^2)$ with attachment conditions at triple junctions, see (3.38b).
$\underline{V}_\partial$	subspace of $\underline{V}$ with restricted direction at boundary intersection points, see (3.38c).
$\nabla_h^i u(\vec{z})$	difference quotient, $= (u(\vec{z} + h\vec{e}_i) - u(\vec{z}))/h$, $i = 1, 2$.
q_j^i	$i = 1, \ldots, N_C$, $j = 1, \ldots, N_i$, discrete points in $[0, 1]$, $\vec{X}_i^m(q_j^i)$ are corresponding mesh points (nodes) of polygonal curves Γ^m.
j_0^i	$= 0$ if $\partial\Gamma_i^m \neq \emptyset$, and $= 1$ else.

W^h	discrete subspace of W, contains piecewise linear, scalar-valued functions, see Section 3.3.1.
$\underline{V}^h$	discrete subspace of $\underline{V}$, contains piecewise linear, vector-valued functions, see Section 3.3.1.
$\chi_{i,j}$	$i = 1, \dots, N_C$, $j = j_0^i, \dots, N_i$, standard basis of W^h.
$\Gamma^m = (\Gamma_1^m, \dots, \Gamma_{N_C}^m)$	polygonal curve, approximation of $\Gamma(t = t_m)$, $m = 0, \dots, M$.
$\vec{X}^m = (\vec{X}_1^m, \dots, \vec{X}_{N_C}^m)$	approximation of $\vec{x}(\,.\,, t_m)$.
$\vec{X}_{i,j}^m$	$= \vec{X}_i^m(q_j^i)$, $i = 1, \dots, N_C$, $j = 0, \dots, N_i$.
$\delta\vec{X}^{m+1}$	$= \vec{X}^{m+1} - \vec{X}^m$, change of the discrete curve(s).
$\kappa^m = (\kappa_1^m, \dots, \kappa_{N_C}^m)$	approximation of $\kappa(\,.\,, t_m)$.
$\kappa_{i,j}^m$	$= \kappa_i^m(q_j^i)$, $i = 1, \dots, N_C$, $j = 0, \dots, N_i$.
$\underline{V}_\partial^h(\vec{X}^m)$	subspace of $\underline{V}^h$ with 90° angle condition at boundary intersection points, see Section 3.3.1.
$\langle u, v\rangle_m$	L^2-inner product for $u = (u_1, \dots, u_{N_C}), v = (v_1, \dots, v_{N_C}) \in L^2(I_1, \mathbb{R}^{(2)}) \times \dots \times L^2(I_{N_C}, \mathbb{R}^{(2)})$, integration over polygonal curves $\Gamma_1^m, \dots, \Gamma_{N_C}^m$, see Section 3.3.1.
$\langle u, v\rangle_m^h$	mass lumped inner product for piecewise continuous functions, see Section 3.3.1.
$h_{i,j-\frac{1}{2}}^m$	$= \|\vec{X}_{i,j}^m - \vec{X}_{i,j-1}^m\| > 0$, distance between two neighbor mesh points.
h^m	$= \max_{i=1,\dots,N_C, j=1,\dots,N_i} h_{i,j-\frac{1}{2}}^m$, maximum distance between two neighbor nodes.
$\vec{\nu}^m = (\vec{\nu}_1^m, \dots, \vec{\nu}_{N_C}^m)$	discrete normal vector field on Γ^m, see Section 3.3.1.
$\vec{\nu}_{i,j-\frac{1}{2}}^m$	$= (\vec{\nu}_i^m)_{\|[q_{j-1}^i, q_j^i]}$, $i = 1, \dots, N_C$, $j = 1, \dots, N_i$.
$\vec{\omega}^m = (\vec{\omega}_1^m, \dots, \vec{\omega}_{N_C}^m)$	weighted, approximating normal vector field, see Section 3.3.1.
$\vec{\omega}_{i,j}^m$	$= (\vec{\omega}_i^m)(q_j^i)$, $i = 1, \dots, N_C$, $j = 1, \dots, N_i$.
$\mathbb{X}$	Euclidean space associated with $\underline{V}_\partial^h(\vec{X}^m)$, see Section 3.3.2.
Ω^h	spatial discretization of the image domain Ω.
Ω_k^h	$= \Omega^h \cap \overline{\Omega_k^m}$, $k = 1, \dots, N_R$.
α	weighting parameter in the automatic setting of the parameter λ for image denoising, cf. Section 3.3.7.

Notation for curves on surfaces in $\mathbb{R}^3$

$\mathcal{M} \subset \mathbb{R}^3$	two-dimensional surface in $\mathbb{R}^3$. Image domain for images on surfaces.
$T_{\vec{p}}\mathcal{M}$	tangent space of a surface $\mathcal{M}$ at $\vec{p} \in \mathcal{M}$.
$N_{\vec{p}}\mathcal{M}$	$= (T_{\vec{p}}\mathcal{M})^{\perp}$, normal space of a surface $\mathcal{M}$ at $\vec{p} \in \mathcal{M}$.
$N\mathcal{M}$	$= \big\{(\vec{p}, \vec{n}) : \vec{p} \in \mathcal{M},\, \vec{n} \in N_{\vec{p}}\mathcal{M}\big\}$, normal bundle.
Γ	curve on $\mathcal{M}$ or union of curves.
N_R	number of regions.
N_C	number of curves.
N_T	number of triple junctions.
$\kappa_{\mathcal{M}}$, κ_{Φ}	geodesic and normal curvature of a curve Γ on a surface $\mathcal{M}$.
$\vec{x}$	parameterization of the curve (or set of curves) Γ.
I_i	one-dimensional reference manifold of a curve Γ_i, $i = 1, \ldots, N_C$, such that the curve can be parameterized by an $\vec{x}_i : I_i \to \mathbb{R}^3$.
$\vec{n}_{\Phi}$	normal vector field on a surface $\mathcal{M}$.
$\vec{\nu}_{\Phi}$	vector field along $\Gamma \subset \mathcal{M}$, normal to the surface $\mathcal{M}$, see Definition 2.27.
$\vec{\nu}_{\mathcal{M}}$	vector field along $\Gamma \subset \mathcal{M}$, tangent to the surface $\mathcal{M}$, normal to the curve Γ, see Definition 2.27.
V_n	velocity of an evolving curve in direction $\vec{\nu}_{\mathcal{M}}$.
$\int_{\Gamma} u \,.\, v \,\mathrm{d}s$	$= \sum_{i=1}^{N_C} \int_{I_i} u_i \,.\, v_i \, \|(\vec{x}_i)_{\rho}\| \,\mathrm{d}\rho$ for scalar or vector-valued functions $u = (u_1, \ldots, u_{N_C}), v = (v_1, \ldots, v_{N_C}) \in L^2(I_1, \mathbb{R}^{(3)}) \times \ldots \times L^2(I_{N_C}, \mathbb{R}^{(3)})$.
$\underline{V}_{\Phi}$	$= \big\{\vec{\eta} : I \to \mathbb{R}^3 : \vec{\eta}$ is smooth and $\vec{\eta} \,.\, \vec{\nu}_{\phi} = 0\big\}$.
$\vec{\Lambda}_k$	triple junctions, $k = 1, \ldots, N_T$.
$\vec{P}_{\mathcal{M}}$	projection onto the part in direction $\vec{\nu}_{\mathcal{M}}$, see Section 4.2.1.
$(\vec{\eta}, \vec{\chi})_{2,\mathcal{M},nor}$	$= \int_{\Gamma} \vec{P}_{\mathcal{M}}\vec{\eta} \,.\, \vec{P}_{\mathcal{M}}\vec{\chi} \,\mathrm{d}s$, an inner product for $\vec{\eta}, \vec{\chi} : I \to \mathbb{R}^3$.
$\nabla_{\mathcal{M}} f$	surface gradient of $f : \mathcal{M} \to \mathbb{R}$.
$\Delta_{\mathcal{M}} f$	Laplace-Beltrami operator of $f : \mathcal{M} \to \mathbb{R}$.
W	$= H^1(I_1, \mathbb{R}) \times \ldots \times H^1(I_{N_C}, \mathbb{R})$.
$\underline{V}$	certain subspace of $H^1(I_1, \mathbb{R}^3) \times \ldots \times H^1(I_{N_C}, \mathbb{R}^3)$ with attachment conditions at triple junctions, see (4.23).
$\underline{V}_{\Phi}$	subspace of $\underline{V}$, where elements are orthogonal to $\vec{n}_{\Phi}$.
q_j^i	$i = 1, \ldots, N_C$, $j = 1, \ldots, N_i$, discrete points in $[0, 1]$, $\vec{X}_i^m(q_j^i)$ are corresponding mesh points (nodes) of polygonal curves Γ^m.
j_0^i	$= 0$ if $\partial\Gamma_i^m \neq \emptyset$, and $= 1$ else.
W^h	discrete subspace of W, contains piecewise linear, scalar-valued functions, see Section 4.3.1.
$\underline{V}^h$	discrete subspace of $\underline{V}$, contains piecewise linear, vector-valued functions, see Section 4.3.1.

$\chi_{i,j}$	$i = 1, \ldots, N_C$, $j = j_0^i, \ldots, N_i$, standard basis of W^h.
$\Gamma^m = (\Gamma_1^m, \ldots, \Gamma_{N_C}^m)$	polygonal curve, approximation of $\Gamma(t = t_m)$, $m = 0, \ldots, M$.
$\vec{X}^m$	approximation of $\vec{x}(\,.\,, t_m)$.
$\delta\vec{X}^{m+1}$	$= \vec{X}^{m+1} - \vec{X}^m$, change of the discrete curve(s).
$\kappa_{\mathcal{M}}^m$	approximation of $\kappa_{\mathcal{M}}(\,.\,, t_m)$.
$\langle u, v\rangle_m$	L^2-inner product for $u = (u_1, \ldots, u_{N_C}), v = (v_1, \ldots, v_{N_C}) \in L^2(I_1, \mathbb{R}^{(3)}) \times \ldots \times L^2(I_{N_C}, \mathbb{R}^{(3)})$, integration over polygonal curves $\Gamma_1^m, \ldots, \Gamma_{N_C}^m$, see Section 4.3.1.
$\langle u, v\rangle_m^h$	mass lumped inner product for piecewise continuous functions, see Section 4.3.1.
$\vec{\omega}_\Phi^m$	$= (\vec{\omega}_{\Phi,1}^m, \ldots, \vec{\omega}_{\Phi,N_C}^m)$, vector field defined by $\vec{\omega}_{\Phi,i}^m(q_j^i) = \vec{n}_\Phi(\vec{X}_i^m(q_j^i))$.
$\vec{\omega}_d^m$	$= (\vec{\omega}_{d,1}^m, \ldots, \vec{\omega}_{d,N_C}^m)$, locally showing in tangential direction w.r.t. Γ_i^m.
$\vec{\omega}_{\mathcal{M}}^m$	$= (\vec{\omega}_{\mathcal{M},1}^m, \ldots, \vec{\omega}_{\mathcal{M},N_C}^m)$, given by cross-products: $\vec{\omega}_{\mathcal{M},i}^m(q_j^i) = \vec{\omega}_{d,i}^m(q_j^i) \times \vec{\omega}_{\Phi,i}^m(q_j^i)$.
$\underline{V}_\Phi^h$	$= \left\{ \vec{\eta} \in \underline{V}^h \,:\, \vec{\eta}_i \,.\, \vec{\omega}_{\Phi,i}^m = 0, \quad i = 1, \ldots, N_C \right\}$.
$\mathbb{X}$	Euclidean space associated with $\underline{V}^h$, see Section 4.3.2.
$\mathbb{X}_\Phi$	Euclidean space associated with $\underline{V}_\Phi^h$, see Section 4.3.2.
$\mathcal{T}^h$	triangulation of a polyhedral surface $\mathcal{M}$.
σ^h	simplices / triangles belonging to $\mathcal{T}^h$.
$\lvert\sigma^h\rvert$	area of a simplex σ^h.
$\vec{p}_{\sigma^h,j}$	vertices of σ^h, $j = 1, 2, 3$.
Ω_k^h	$= \Omega_k^M$ for $k = 1, \ldots, N_R$.
$\mathcal{T}_k^h$	$= \{\sigma^h \in \mathcal{T}^h \,:\, \sigma^h \subset \Omega_k^h\}$.
S_k^h	finite element space of piecewise linear functions defined on $\overline{\Omega_k^h}$, see Section 4.3.6.
$\langle u^h, v^h\rangle^h$	$= \frac{1}{3} \sum_{\sigma^h \in \mathcal{T}_k^h} \lvert\sigma^h\rvert \sum_{j=1}^3 (u^h\, v^h)((\vec{p}_{\sigma^h,j})^-)$, see Section 4.3.6.

Notation for surfaces in $\mathbb{R}^3$

$\Omega \subset \mathbb{R}^3$	three-dimensional image domain.
Γ	two-dimensional surface or union of surfaces in Ω.
N_R	number of regions.
N_S	number of surfaces.
κ	mean curvature of Γ (sum of principal curvatures).
$\vec{\kappa} = \kappa\vec{\nu}$	mean curvature vector.
$\vec{x}$	parameterization of the surface (or set of surfaces) Γ.
Υ_i	reference manifold of a surface Γ_i, $i = 1, \ldots, N_S$, such that the surface can be parameterized by an $\vec{x}_i : \Upsilon_i \to \mathbb{R}^3$.
$\vec{\nu}$	a normal vector field on the surface(s) Γ.
V_n	velocity of an evolving surface in direction $\vec{\nu}$ (normal velocity).
$g_{kl}^{(i)}(\vec{q})$	$= \frac{\partial \vec{x}_i}{\partial q_k}(\vec{q}) \,.\, \frac{\partial \vec{x}_i}{\partial q_l}(\vec{q})$ for $\vec{q} \in \Upsilon_i$.
$\int_\Gamma u \,.\, v \,\mathrm{d}A$	$= \sum_{i=1}^{N_S} \int_{\Upsilon_i} (u_i \,.\, v_i) \circ \vec{x}_i \sqrt{\det(g_{kl}^{(i)})} \,\mathrm{d}q$ for $u, v \in L^2(\Upsilon_1, \mathbb{R}^{(3)}) \times \ldots \times L^2(\Upsilon_{N_S}, \mathbb{R}^{(3)})$.
W	$= H^1(\Upsilon_1, \mathbb{R}) \times \ldots \times H^1(\Upsilon_{N_S}, \mathbb{R})$.
$\underline{V}$	$= H^1(\Upsilon_1, \mathbb{R}^3) \times \ldots \times H^1(\Upsilon_{N_S}, \mathbb{R}^3)$.
$\Gamma^m = (\Gamma_1^m, \ldots, \Gamma_{N_C}^m)$	polyhedral surface, approximation of $\Gamma(t = t_m)$, $m = 0, \ldots, M$.
$N_{i,F}$	number of faces/simplices belonging to Γ_i^m, $i = 1, \ldots, N_S$.
$N_{i,V}$	number of vertices/nodes belonging to Γ_i^m, $i = 1, \ldots, N_S$.
$N_{i,E}$	number of edges belonging to Γ_i^m, $i = 1, \ldots, N_S$.
$\sigma_{i,j}^m$	simplices belonging to Γ_i^m, $i = 1, \ldots, N_S$, $j = 1, \ldots, N_{i,F}$.
g_i	genus of the surface Γ_i^m.
$\vec{X}^m$	$= (\vec{X}_1^m, \ldots, \vec{X}_{N_S}^m)$, approximation of $\vec{x}(\,.\,, t_m)$.
$\delta\vec{X}^{m+1}$	$= \vec{X}^{m+1} - \vec{X}^m$, change of the triangulated surfaces.
κ^m	$= (\kappa_1^m, \ldots, \kappa_{N_S}^m)$, approximation of $\kappa(\,.\,, t_m)$.
$\vec{q}_{i,j}^m$	vertices of Γ_i^m, $i = 1, \ldots, N_S$, $j = 1, \ldots, N_{i,V}$.
h	$= \max_{i=1,\ldots,N_S, j=1,\ldots,N_{i,F}} \mathrm{diam}(\sigma_{i,j}^m)$, maximum diameter of a simplex.
$W(\Gamma^m)$	finite element space, consisting of piecewise linear, scalar-valued functions, see Section 5.3.1.
$\underline{V}(\Gamma^m)$	finite element space, consisting of piecewise linear, vector-valued functions, see Section 5.3.1.
$\chi_{i,j}^m$	basis functions of $W(\Gamma^m)$, $i = 1, \ldots, N_S$, $j = 1, \ldots, N_{i,V}$.
$\langle u, v \rangle_m$	L^2-inner product over the current polyhedral surfaces Γ^m, defined for $u = (u_1, \ldots, u_{N_S})$, $v = (v_1, \ldots, v_{N_S}) \in L^2(\Gamma_1^m, \mathbb{R}^{(3)}) \times \ldots \times L^2(\Gamma_{N_S}^m, \mathbb{R}^{(3)})$, see Section 5.3.1.
$\langle u, v \rangle_m^h$	mass lumped inner product for piecwiese continuous functions u, v with possible jumps across the edges of simplices.

$\vec{q}^m_{i,j_l}$	vertices of $\sigma^m_{i,j}$, $i = 1,\dots,N_S$, $j = 1,\dots,N_{i,F}$, $j_l \in \{1,\dots,N_{i,V}\}$ for $l = 1,2,3$.
$\lvert\sigma^m_{i,j}\rvert$	area of the simplex $\sigma^m_{i,j}$.
$\vec{\nu}^m$	$= (\vec{\nu}^m_1,\dots,\vec{\nu}^m_{N_S})$, discrete normal vector field on Γ^m, see Section 5.3.1.
$\vec{\nu}^m_{i,j}$	$= \vec{\nu}^m_i\lvert_{\sigma^m_{i,j}}$, for $i = 1,\dots,N_S$, $j = 1,\dots,N_{i,F}$.
$\vec{\omega}^m$	$= (\vec{\omega}^m_1,\dots,\vec{\omega}^m_{N_S})$, weighted, approximating normal vector field, see Section 5.3.1.
$\vec{\omega}^m_{i,j}$	$= (\vec{\omega}^m_i)(q^i_j)$, $i = 1,\dots,N_S$, $j = 1,\dots,N_{i,V}$.
$\mathcal{T}^m_{i,j}$	$= \left\{\sigma^m_{i,l} \,:\, \vec{q}^m_{i,j} \in \overline{\sigma^m_{i,l}}\right\}$.
$\Lambda^m_{i,j}$	$= \bigcup_{\sigma^m_{i,l}\in\mathcal{T}^m_{i,j}} \overline{\sigma^m_{i,l}}$.
$\vec{v}^m$	$= (\vec{v}^m_1,\dots,\vec{v}^m_{N_S})$, discrete normal vector field, see Section 5.3.1.
$\vec{v}^m_{i,j}$	$= \vec{v}^m_i(\vec{q}^m_{i,j}) = \vec{\omega}^m_{i,j}/\lVert\vec{\omega}^m_{i,j}\rVert$.
$\vec{\tau}_1$, $\vec{\tau}_2$	normal vector fields such that $\{\vec{v}^m_{i,j}, \vec{\tau}^m_{1,i,j}, \vec{\tau}^m_{2,i,j}\}$ is an orthonormal basis of $\mathbb{R}^3$.
$\delta X^{\min}_n > 0$, $\delta X^{\max}_n > 0$, λ_t	parameters for the time step control, cf. Section 5.3.4.
$A_{\text{desired}} > 0$, $a > 0$	parameters for the refinement of simplices, cf. Section 5.3.4.
$\alpha_0 \geq 0$	parameter for mesh regularization via induced tangential motion, cf. Section 5.3.5.

Bibliography

Adalsteinsson, D. and Sethian, J. A. (1995), A Fast Level Set Method for Propagating Interfaces, *Journal of Computational Physics*, 118(2):269–277.

Ambrosio, L. (1990), Existence theory for a new class of variational problems, *Archive for Rational Mechanics and Analysis*, 111(4):291–322.

Ambrosio, L. and Tortorelli, V. M. (1990), Approximation of Functional Depending on Jumps by Elliptic Functionals via Γ-Convergence, *Communications on Pure and Applied Mathematics*, 43(8):999–1036.

Araki, S., Yokoya, N., Iwasa, H., and Takemura, H. (1997), Splitting of active contour models based on crossing detection for extraction of multiple objects, *Systems and Computers in Japan*, 28(11):34–42.

Arbelaez, P., Maire, M., Fowlkes, C., and Malik, J. (2011), Contour detection and hierarchical image segmentation, *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5):898–916.

Ardon, R., Cohen, L. D., and Yezzi, A. (2005), A New Implicit Method for Surface Segmentation by Minimal Paths: Applications in 3D Medical Images. In Rangarajan, A., Vemuri, B., and Yuille, A. L., editors, *Energy Minimization Methods in Computer Vision and Pattern Recognition*, volume 3757 of *Lecture Notes in Computer Science*, pages 520–535. Springer Berlin Heidelberg.

Armato, III, S. G., McLennan, G., Bidaut, L., McNitt-Gray, M. F., Meyer, C. R., Reeves, A. P., Zhao, B., Aberle, D. R., Henschke, C. I., Hoffman, E. A., Kazerooni, E. A., MacMahon, H., van Beek, E. J. R., Yankelevitz, D., Biancardi, A. M., Bland, P. H., Brown, M. S., Engelmann, R. M., Laderach, G. E., Max, D., Pais, R. C., Qing, D. P.-Y., Roberts, R. Y., Smith, A. R., Starkey, A., Batra, P., Caligiuri, P., Farooqi, A., Gladish, G. W., Jude, C. M., Munden, R. F., Petkovska, I., Quint, L. E., Schwartz, L. H., Sundaram, B., Dodd, L. E., Fenimore, C., Gur, D., Petrick, N., Freymann, J., Kirby, J., Hughes, B., Vande Casteele, A., Gupte, S., Sallam, M., Heath, M. D., Kuhn, M. H., Dharaiya, E., Burns, R., Fryd, D. S., Salganicoff, M., Anand, V., Shreter, U., Vastagh, S., Croft, B. Y., and Clarke, L. P. (2011), The Lung Image Database Consortium (LIDC) and Image Database Resource Initiative (IDRI): A Completed Reference Database of Lung Nodules on CT Scans, *Medical Physics*, 38(2):915–931.

Astos Solutions GmbH (2013). Camera Simulator 1.2.

Aubert, G. and Kornprobst, P. (2006), *Mathematical Problems in Image Processing*. Springer, New York.

Aujol, J.-F. and Kang, S. H. (2006), Color image decomposition and restoration, *Journal of Visual Communication and Image Representation*, 17(4):916–928.

Balažovjech, M., Mikula, K., Petrášová, M., and Urbán, J. (2012), Lagrangean method with topological changes for numerical modelling of forest fire propagation. In *Proceedings of ALGORITMY 2012, 19th Conference on Scientific Computing*, pages 42–52, Vysoké Tatry, Podbansk'v, Slovakia.

Bänsch, E., Morin, P., and Nochetto, R. H. (2005), Finite Element Methods for Surface Diffusion: the Parametric Case, *Journal of Computational Physics*, 203(1):321–343.

Barrett, J. W., Garcke, H., and Nürnberg, R. (2007a), On the variational approximation of combined second and fourth order geometric evolution equations, *SIAM Journal on Scientific Computing*, 29(3):1006–1041.

Barrett, J. W., Garcke, H., and Nürnberg, R. (2007b), A parametric finite element method for fourth order geometric evolution equations, *Journal of Computational Physics*, 222(1): 441–467.

Barrett, J. W., Garcke, H., and Nürnberg, R. (2008a), A variational formulation of anisotropic geometric evolution equations in higher dimensions, *Numerische Mathematik*, 109(1):1–44.

Barrett, J. W., Garcke, H., and Nürnberg, R. (2008b), On the parametric finite element approximation of evolving hypersurfaces in $\mathbb{R}^3$, *Journal of Computational Physics*, 227(9): 4281–4307.

Barrett, J. W., Garcke, H., and Nürnberg, R. (2008c), Parametric approximation of Willmore flow and related geometric evolution equations, *SIAM Journal on Scientific Computing*, 31 (1):225–253.

Barrett, J. W., Garcke, H., and Nürnberg, R. (2010a), Numerical Approximation of Gradient Flows for Closed Curves in $\mathbb{R}^d$, *IMA Journal of Numerical Analysis*, 30(1):4–60.

Barrett, J. W., Garcke, H., and Nürnberg, R. (2010b), Parametric Approximation of Surface Clusters Driven by Isotropic and Anisotropic Surface Energies, *Interfaces and Free Boundaries*, 12(2):187–234.

Barrett, J. W., Garcke, H., and Nürnberg, R. (2014), Phase field models versus parametric front tracking methods: Are they accurate and computationally efficient?, *Communications in Computational Physics*, 15:506–555.

Beneš, M., Chalupecký, V., and Mikula, K. (2004), Geometrical Image Segmentation by the Allen-Cahn Equation, *Applied Numerical Mathematics*, 51:187–205.

Beneš, M., Kimura, M., Pauš, P., Ševčovič, D., Tsujikawa, T., and Yazaki, S. (2008), Application of a Curvature Adjusted Method in Image Segmentation, *Bulletin of the Institute of Mathematics, Academia Sinica (New Series)*, 3(4):509–523.

Benninghoff, H. and Garcke, H. (2014), Efficient Image Segmentation and Restoration Using Parametric Curve Evolution With Junctions and Topology Changes, *SIAM Journal on Imaging Sciences*, 7(3):1451–1483.

Benninghoff, H., Boge, T., and Rems, F. (2014), Autonomous Navigation for On-Orbit Servicing, *KI - Künstliche Intelligenz*, 28(2):77–83.

Bertalmio, M., Sapiro, G., Caselles, V., and Ballester, C. (2000), Image Inpainting. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, pages 417–424.

Bertalmio, M., Cheng, L.-T., Osher, S., and Sapiro, G. (2001), Variational Problems and Partial Differential Equations on Implicit Surfaces: The Framework and Examples in Image Processing and Pattern Formation, *Journal of Computational Physics*, 174(2):759–780.

Bezdek, J. C., Keller, J., Krisnapuram, R., and Pal, N. R. (2005), *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing*. The Handbooks of Fuzzy Sets Series. Springer.

Bhaskaran, S., Riedel, J. E., and Synnott, S. P. (1996), Autonomous optical navigation for interplanetary missions. In *Proceedings SPIE 2810, Space Sciencecraft Control and Tracking in the New Millennium*, volume 2810, pages 32–43.

Black, M. J., Sapiro, G., Marimont, D. H., and Heeger, D. (1998), Robust Anisotropic Diffusion, *IEEE Transactions on Image Processing*, 7(3):421–432.

Bonin-Font, F., Ortiz, A., and Oliver, G. (2008), Visual navigation for mobile robots: A survey, *Journal of Intelligent and Robotic Systems*, 53(3):263–296.

Brakke, K. A. (1992), The Surface Evolver, *Experimental Mathematics*, 1(2):141–165.

Bresson, X., Esedoglu, S., Vandergheynst, P., Thiran, J., and Osher, S. (2007), Fast Global Minimization of the Active Contour/Snake Model, *Journal of Mathematical Imaging and Vision*, 28(2):151–167.

Brochu, T. and Bridson, R. (2009), Robust Topological Operations for Dynamic Explicit Surfaces, *SIAM Journal on Scientific Computing*, 31(4):2472–2493.

Brook, A., Kimmel, R., and Sochen, N. A. (2003), Variational Restoration and Edge Detection for Color Images, *Journal of Mathematical Imaging and Vision*, 18(3):247–268.

Caselles, V., Kimmel, R., and Sapiro, G. (1997a), Geodesic Active Contours, *International Journal of Computer Vision*, 22(1):61–79.

Caselles, V., Kimmel, R., Sapiro, G., and Sbert, C. (1997b), Minimal surfaces: a geometric three dimensional segmentation approach, *Numerische Mathematik*, 77:423–451.

Chambolle, A., Cremers, D., and Pock, T. (2012), A Convex Approach to Minimal Partitions, *SIAM Journal on Imaging Sciences*, 5(4):1113–1158.

Chan, T. F. and Vese, L. A. (2001), Active Contours Without Edges, *IEEE Transactions on Image Processing*, 10(2):266–277.

Chan, T. F., Sandberg, B. Y., and Vese, L. A. (2000), Active Contours without Edges for Vector-Valued Images, *Journal of Visual Communication and Image Representation*, 11 (2):130–141.

Chan, T. F., Kang, S. H., and Shen, J. (2001), Total Variation Denoising and Enhancement of Color Images Based on the CB and HSV Color Models, *Journal of Visual Communication and Image Representation*, 12(4):422–435.

Chan, T. F., Kang, S. H., and Shen, J. (2002), Euler's Elastica And Curvature Based Inpaintings, *SIAM Journal on Applied Mathematics*, 63:564–592.

Chan, T. F., Esedoglu, S., and Nikolova, M. (2006), Algorithms for Finding Global Minimizers of Image Segmentation and Denoising Models, *SIAM Journal on Applied Mathematics*, 66: 1632–1648.

Chandola, V., Banerjee, A., and Kumar, V. (2009), Anomaly Detection: A Survey, *ACM Computing Surveys*, 41(3).

Chen, L. (2004), Mesh Smoothing Schemes Based on Optimal Delaunay Triangulations. In *13th International Meshing Roundtable, Sandia National Laboratories*, Williamsburg, VA.

Chen, L. and Holst, M. (2011), Efficient Mesh Optimization Schemes based on Optimal Delaunay Triangulations, *Computer Methods in Applied Mechanics and Engineering*, 200: 967–984.

Cheng, L.-T., Burchard, P., Merriman, B., and Osher, S. (2002), Motion of Curves Constrained on Surfaces Using a Level-Set Approach, *Journal of Computational Physics*, 175 (2):604–644.

Chin, R. T. and Harlow, C. A. (1982), Automated Visual Inspection: A Survey, *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-4(6):557–573.

Chung, G. and Vese, L. A. (2009), Image segmentation using a multilayer level-set approach, *Computing and Visualization in Science*, 12(6):267–285.

Cohen, L. D. (1991), On Active Contour Models and Balloons, *Computer Vision, Graphics, and Image Processing: Image Understanding*, 53(2):211–218.

Cohen, L. D. and Cohen, I. (1993), Finite Element Methods for Active Contour Models and Balloons for 2D and 3D Images, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):1131–1147.

Cremers, D., Schnörr, C., and Weickert, J. (2001), Diffusion-snakes: Combining statistical shape knowledge and image information in a variational framework. In *Proceedings of the IEEE Workshop on Variational and Level Set Methods in Computer Vision*, pages 137–144, Vancouver.

Čunderlík, R., Mikula, K., and Tunega, M. (2013), Nonlinear diffusion filtering of data on the Earth's surface, *Journal of Geodesy*, 87(2):143–160.

Davis, T. (2004), Algorithm 832: UMFPACK, an unsymmetric-pattern multifrontal method, *ACM Transactions on Mathematical Software*, 30(2):196–199.

Deckelnick, K., Dziuk, G., and Elliott, C. M. (2005), Computation of Geometric Partial Differential Equations and Mean Curvature Flow, *Acta Numerica*, 14:139–232.

DeSouza, G. N. and Kak, A. C. (2002), Vision for mobile robot navigation: a survey, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(2):237–267.

Dobrokhodov, V. N., Kaminer, I. I., Jones, K. D., and Ghabcheloo, R. (2006), Vision-based tracking and motion estimation for moving targets using small UAVs. In *Proceedings of American Control Conference*.

Doğan, G., Morin, P., and Nochetto, R. H. (2008), A Variational Shape Optimization Approach for Image Segmentation with a Mumford-Shah Functional, *SIAM Journal on Scientific Computing*, 30(6):3028–3049.

Du, C.-J. and Sun, D.-W. (2004), Recent developments in the applications of image processing techniques for food quality evaluation, *Trends in Food Science & Technology*, 15(5):230–249.

Dubrovina, A., Rosman, G., and Kimmel, R. (2013), Active Contours for Multi-region Image Segmentation with a Single Level Set Function. In Kuijper, A., Bredies, K., Pock, T., and Bischof, H., editors, *Scale Space and Variational Methods in Computer Vision*, volume 7893 of *Lecture Notes in Computer Science*, pages 416–427. Springer Berlin Heidelberg.

Dziuk, G. (1991), An algorithm for evolutionary surfaces, *Numerische Mathematik*, 58(1): 603–611.

Dziuk, G. and Elliott, C. M. (2013), Finite Element Methods for Surface PDEs, *Acta Numerica*, 22:289–396.

Eck, C., Garcke, H., and Knabner, P. (2008), *Mathematische Modellierung*. Springer, Berlin, Heidelberg, 1st edition.

Eschenburg, J.-H. and Jost, J. (2007), *Differentialgeometrie und Minimalflächen*. Springer, Berlin, Heidelberg, 2nd edition.

Fei-Fei, L., Fergus, R., and Perona, P. (2004), Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories. In *Proceedings of the IEEE, CVPR 2004 Workshop on Generative-Model Based Vision*.

Fergus, R., Perona, P., and Zisserman, A. (2003), Object Class Recognition by Unsupervised Scale-Invariant Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 264–271.

Gage, M. and Hamilton, R. S. (1986), The Heat Equation Shrinking Convex Plane Curves, *Journal of Differential Geometry*, 23:69–96.

Garcke, H. (2013), Curvature Driven Interface Evolution, *Jahresbericht der Deutschen Mathematiker-Vereinigung*, 115(2):63–100.

Garcke, H. and Wieland, S. (2006), Surfactant Spreading on Thin Viscous Films: Nonnegative Solutions of a Coupled Degenerate System, *SIAM Journal on Mathematical Analysis*, 37 (6):2025–2048.

Gavrila, D. and Philomin, V. (1999), Real-time object detection for smart vehicles. In *Proceedings of the 7th IEEE International Conference on Computer Vision*, volume 1, pages 87–93.

Giga, Y. (2006), *Surface Evolution Equations: A Level Set Approach*. Monographs in Mathematics. Springer, London.

Gonzalez, R. C. and Woods, R. E. (2001), *Digital Image Processing*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2nd edition.

Grayson, M. A. (1987), The Heat Equation Shrinks Embedded Plane Curves to Round Points, *Journal of Differential Geometry*, 26:285–314.

Hintermüller, M. and Ring, W. (2004), An Inexact Newton-CG-Type Active Contour Approach for the Minimization of the Mumford-Shah Functional, *Journal of Mathematical Imaging and Vision*, 20(1-2):19–42.

Huisken, G. (1984), Flow by mean curvature of convex surfaces into spheres, *Journal of Differential Geometry*, 20(1):237–266.

Ilmanen, T. (1998). Lectures on Mean Curvature Flow and Related Equations. URL `www.math.ethz.ch/~ilmanen/papers/notes.pdf`.

Jänich, K. (2005), *Vektoranalysis*. Undergraduate texts in mathematics. Springer, Berlin, Heidelberg, New York, 5th edition.

Jensen, J. R. (1996), *Introductory Digital Image Processing: A Remote Sensing Perspective*. Prentice Hall series in geographic information science. Prentice Hall.

Kass, M., Witkin, A., and Terzopoulos, D. (1988), Snakes: Active Contour Models, *International Journal of Computer Vision*, 1(4):321–331.

Kichenassamy, S., Kumar, A., Olver, P., Tannenbaum, A., and Yezzi Jr., A. (1996), Conformal Curvature Flows: From Phase Transitions to Active Vision, *Archive for Rational Mechanics and Analysis*, 134(3):275–301.

Kimmel, R. (1997), Intrinsic Scale Space for Images on Surfaces: The Geodesic Curvature Flow, *Graphical Models and Image Processing*, 59(5):365–372.

Kimmel, R. and Bruckstein, A. M. (2003), Regularized Laplacian Zero Crossings as Optimal Edge Integrators, *International Journal of Computer Vision*, 53(3):225–243.

Koenderink, J. J. (1984), The structure of images, *Biological Cybernetics*, 50(5):363–370.

Krüger, M., Delmas, P., and Gimelfarb, G. (2008), Active contour based segmentation of 3D surfaces. In *Proceedings of the European Conference on Computer Vision*, pages 350–363, Marseille, France.

Kuhn, H. W. (1955), The Hungarian method for the assignment problem, *Naval Research Logistic Quarterly*, 2:83–97.

Kühnel, W. (2005), *Differentialgeometrie*. Vieweg-Studium: Aufbaukurs Mathematik. Vieweg, 4th edition.

Lai, R. and Chan, T. F. (2011), A Framework for Intrinsic Image Processing on Surfaces, *Computer Vision and Image Understanding*, 115(12):1647–1661.

Lang, S. (2002), *Introduction to Differentiable Manifolds*. Universitext. Springer, New York, Berlin, Heidelberg, 2nd edition.

Lee, J. M. (2002), *Introduction to Smooth Manifolds*, volume 218 of *Graduate Texts in Mathematics*. Springer, New York, Heidelberg, Dordrecht, London.

Li, H., Yezzi, A., and Cohen, L. D. (2005), Fast 3D Brain Segmentation Using Dual-Front Active Contours with Optional User-Interaction. In Liu, Y., Jiang, T., and Zhang, C., editors, *Computer Vision for Biomedical Image Applications*, volume 3765 of *Lecture Notes in Computer Science*, pages 335–345. Springer Berlin Heidelberg.

Malladi, R., Sethian, J. A., and Vemuri, B. C. (1995), Shape Modeling with Front Propagation: A Level Set Approach, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(2):158–175.

Markham, B. L. and Barker, J. L. (1985), Spectral characterization of the LANDSAT Thematic Mapper sensors, *International Journal of Remote Sensing*, 6(5):697–716.

Martin, D., Fowlkes, C., Tal, D., and Malik, J. (2001), A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics. In *8th IEEE International Conference on Computer Vision*, volume 2, pages 416–423, Vancouver, Canada.

Mikula, K. and Urbán, J. (2012), New fast and stable Lagrangean method for image segmentation. In *Proceedings of the 5th International Congress on Image and Signal Processing (CISP 2012)*, pages 834–842, Chongquing, China.

Mikula, K. and Ševčovič, D. (2004a), A Direct Method for Solving an Anisotropic Mean Curvature Flow of Plane Surfaces with an External Force, *Mathematical Methods in the Applied Sciences*, 27:1545–1565.

Mikula, K. and Ševčovič, D. (2004b), Computational and Qualitative Aspects of Evolution of Curves Driven by Curvature and External Force, *Computing and Visualization in Science*, 6:211–225.

Mikula, K. and Ševčovič, D. (2006), Evolution of Curves on a Surface Driven by the Geodesic Curvature and External Force, *Applicable Analysis*, 85(4):345–362.

Mikula, K., Peyriéras, N., Remešíková, M., and Stašová (2011), Segmentation of 3D Cell Membrane Images by PDE Methods and its Applications, *Computers in Biology and Medicine*, 41(6):326–339.

Mikula, K., Peyriéras, N., and Špir, R. (2014a), Numerical Algorithm for Tracking Cell Dynamics in 4D Biomedical Images, *accepted for publication in Discrete and Continuous Dynamical Systems - Series S*.

Mikula, K., Remešíková, M., Sarkoci, P., and Ševčovič, D. (2014b), Manifold Evolution with Tangential Redistribution of Points, *SIAM Journal on Scientific Computing*, 36(4):A1384–A1414.

Mille, J. (2009), Narrow band region-based active contours and surfaces for 2D and 3D segmentation, *Computer Vision and Image Understanding*, 113(9):946–965.

Moelich, M. and Chan, T. (2003), Tracking Objects with the Chan-Vese Algorithm. Technical report, Computational Applied Mathematics.

Mumford, D. and Shah, J. (1989), Optimal Approximation by Piecewise Smooth Functions and Associated Variational Problems, *Communications on Pure and Applied Mathematics*, 42:577–685.

Nakhmani, A. and Tannenbaum, A. (2012), Self-crossing detection and location for parametric active contours, *IEEE Transactions on Image Processing*, 21(7):3150–3156.

NASA (2014). NASA Earth Observations. URL `http://neo.sci.gsfc.nasa.gov/`.

Nixon, M. S. and Aguado, A. S. (2002), *Feature Extraction and Image Processing*. Newnes, Oxford, Auckland, Boston, Johannesburg, Melbourne, New Delhi, 1st edition.

Osher, S. and Sethian, J. A. (1988), Fronts Propagating with Curvature Dependent Speed: Algorithms Based on Hamilton-Jacobi Formulations, *Journal of Computational Physics*, 79(1):12–49.

Paragios, N. and Deriche, R. (2000), Geodesic Active Contours and Level Sets for Detection and Tracking of Moving Objects, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(3):266–280.

Paysan, P., Knothe, R., Amberg, B., Romdhani, S., and Vetter, T. (2009), A 3D Face Model for Pose and Illumination Invariant Face Recognition. In *Proceedings of the 6th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS) for*

Security, Safety and Monitoring in Smart Environments, pages 296–301, Genova, Italy. IEEE.

Perona, P. and Malik, J. (1990), Scale-Space and Edge Detection Using Anisotropic Diffusion, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639.

Pock, T., Chambolle, A., Cremers, D., and Bischof, H. (2009a), A convex relaxation approach for computing minimal partitions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2009)*, pages 810–817.

Pock, T., Cremers, D., Bischof, H., and Chambolle, A. (2009b), An Algorithm for Minimizing the Mumford-Shah Functional. In *Proceedings of the 12th IEEE International Conference on Computer Vision (ICCV 2009)*, pages 1133–1140.

Reeves, A. P. and Biancardi, A. M. (2011). The Lung Image Database Consortium (LIDC) Nodule Size Report, Release: 2011-10-27. URL `http://www.via.cornell.edu/lidc/`.

Reeves, A. P., Biancardi, A. M., Apanasovich, T. V., Meyer, C. R., MacMahon, H., van Beek, E. J., Kazerooni, E. A., Yankelevitz, D., McNitt-Gray, M. F., McLennan, G., Armato III, S. G., Henschke, C. I., Aberle, D. R., Croft, B. Y., and Clarke, L. P. (2007), The lung image database consortium (LIDC): A comparison of different size metrics for pulmonary nodule measurements, *Academic Radiology*, 14(12):1475–1485.

Ronfard, R. (1994), Region-based strategies for active contour models, *International Journal of Computer Vision*, 13(2):229–251.

Rousseau, O. and Bourgault, Y. (2009). Heart Segmentation With an Iterative Chan-Vese Algorithm. URL `http://hal.archives-ouvertes.fr/hal-00403627/en/`.

Rudin, L. I., Osher, S., and Fatemi, E. (1992), Nonlinear Total Variation Based Noise Removal Algorithms, *Phys. D*, 60(1-4):259–268.

Sapiro, G. (2006), *Geometric Partial Differential Equations and Image Analysis*. Cambridge University Press, New York.

Sarti, A., Malladi, R., and Sethian, J. A. (2002), Subjective Surfaces: A Geometric Model for Boundary Completion, *International Journal of Computer Vision*, 46(3):201–221.

Saye, R. I. and Sethian, J. A. (2011), The Voronoi Implicit Interface Method for computing multiphase physics, *Proceedings of the National Academy of Sciences*, 108(49):19498–19503.

Schaeffer, H. and Vese, L. (2014), Active Contours with Free Endpoints, *Journal of Mathematical Imaging and Vision*, 49(1):20–36.

Scherzer, O., Grasmair, M., Grossauer, H., Haltmeier, M., and Lenzen, F. (2009), *Variational Methods in Imaging*. Number 167 in Applied Mathematical Sciences. Springer.

Sethian, J. A. (1999), *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science*. Cambridge University Press, 2rd edition.

Ševčovič, D. and Yazaki, S. (2011), Evolution of Plane Curves with a Curvature Adjusted Tangential Velocity, *Japan Journal of Industrial and Applied Mathematics*, 28:413–442.

Sharma, N. and Aggarwal, L. M. (2010), Automated Medical Image Segmentation Techniques, *Journal of Medical Physics*, 35(1):3–14.

Shen, T. and Huang, X. (2009), 3D Medical Image Segmentation by Multiple-Surface Active Volume Models. In Yang, G.-Z., Hawkes, D., Rueckert, D., Noble, A., and Taylor, C., editors, *Medical Image Computing and Computer-Assisted Intervention MICCAI 2009*, volume 5762 of *Lecture Notes in Computer Science*, pages 1059–1066. Springer Berlin Heidelberg.

Spira, A. and Kimmel, R. (2007), Geometric Curve Flows on Parametric Manifolds, *Journal of Computational Physics*, 223:235–249.

Srikrishnan, V., Chaudhuri, S., Roy, S., and Ševčovič, D. (2007), On stabilisation of parametric active contours. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '07*, pages 1–6, Minneapolis.

Tang, B., Sapiro, G., and Caselles, V. (2002), Color Image Enhancement via Chromaticity Diffusion, *IEEE Transactions on Image Processing*, 10(5):701–707.

Tian, L., Macdonald, C. B., and Ruuth, S. J. (2009), Segmentation on surfaces with the closest point method. In *Proceedings of the 16th IEEE international conference on Image processing*, pages 3009–3012, Cairo, Egypt.

Tsai, A., Yezzi, A., and Willsky, A. S. (2001), Curve Evolution Implementation of the Mumford-Shah Functional for Image Segmentation, Denoising, Interpolation and Magnification, *IEEE Transactions on Image Processing*, 10(8):1169–1186.

Tuia, D., Muoz-Mar, J., and Camps-Valls, G. (2012), Remote sensing image segmentation by active queries., *Pattern Recognition*, 45(6):2180–2192.

Turk, G. and Levoy, M. (1994), Zippered polygon meshes from range images. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques (SIGGRAPH '94)*, pages 311–318, New York, NY, USA. ACM.

Udupa, J. K. and Herman, G. T. (1999), *3D Imaging in Medicine*. CRC Press, 2nd edition.

Vese, L. A. and Chan, T. F. (2002), A Multiphase Level Set Framework for Image Segmentation Using the Mumford and Shah Model, *International Journal of Computer Vision*, 50 (3):271–293.

Woffinden, D. C. and Geller, D. K. (2007), Navigating the Road to Autonomous Orbital Rendezvous, *Journal of Spacecraft and Rockets*, 44(4):898–909.

Wu, C. and Tai, X.-C. (2010), Augmented Lagrangian Method Dual Methods, and Split Bregman Iteration for ROF, Vectorial TV, and High Order Models, *SIAM Journal on Imaging Sciences*, 3(3):300–339.

Wu, C., Zhang, J., Duan, Y., and Tai, X.-C. (2012), Augmented Lagrangian Method for Total Variation Based Image Restoration and Segmentation Over Triangulated Surfaces, *Journal of Scientific Computing*, 50(1):145–166.

Yezzi Jr., A., Kichenassamy, S., Kumar, A., Olver, P., and Tannenbaum, A. (1997), A geometric snake model for segmentation of medical imagery, *IEEE Transactions on Medical Imaging*, 16(2):199–209.

Yu, Z. and Bajaj, C. (2002), Anisotropic Vector Diffusion in Image Smoothing. In *Proceedings of International Conference on Image Processing*, pages 828–831.

Zhou, L. and Li, J. (2013), Image Segmentation on Implicit Surface Based on Chan-Vese Model, *Journal of Theoretical and Applied Information Technology*, 48(1):206–209.