Leopold Franzens Universität Innsbruck

---

Dissertation

# Geometrical and Statistical Methods for Segmentation of Topologically Complex Objects in 3D

Dipl.-Math. Jochen Abhau

in partial fulfillment of the requirements
for the academic degree
**Doctor Rerum Naturalium**

submitted to the Faculty of Mathematics,
Computer Science and Physics

May 2010

Supervised by
Univ.-Prof. Dipl.-Ing. Dr. Otmar Scherzer

# Preface

Segmentation is the process of partitioning image data into multiple regions, such that elements of the same region share a certain property. Of particular interest is an image partitioning which separates an object of interest from its background. In this case, a segmentation is given by the boundary (or contour) of the object.
Segmentation plays a crucial role in many medical imaging applications such as locating tumors, measuring tissue volumes or computer-guided surgery. Furthermore, image segmentation is applied in face recognition, optical character recognition (OCR) and machine vision, to mention but a few applications. As a typical example from medical imaging, a 2D slice of an originally 3D ultrasound image of a cyst and a segmentation is shown in Figure 0.1.
There are various segmentation methods like thresholding, region growing or clustering which produce good results on clear, uncorrupted images, but fail to work if image data is noisy, has low contrast or if the object is even partially occluded. In these situations, energy minimization methods and deformable models have proven very succesful. There, for given image data $I_0$, a real valued functional of type

$$E(C, I_0) \tag{0.1}$$

is mininimized over a certain class of contours $C$. This way, segmentation can be formulated in a more mathematical way: Every (local) minimizer of a functional of type (0.1) is a segmentation of $I_0$. Of course, this definition depends on the choice of $E$. A common choice is

$$E(C, I_0) = E_{DATA}(C, I_0) + E_{SMOOTH}(C). \tag{0.2}$$

In this functional, the fit-to-data term $E_{DATA}$ is constructed, such that $C$ approximates the contour of the object if $E_{DATA}(C, I_0)$ is small. Vice versa $E_{SMOOTH}(C)$ penalizes for smooth contours $C$. By this approach, the segmenting contour $C$ allows for outliers that are frequently present in noisy image data.
Smoothness of the segmenting contour can be regarded as special a priori knowledge on the object to segment. If more generally, a priori knowledge on the shape of the object is available, a template shape $C_T$ is modelled statistically, and functional (0.2) is generalized to

$$E(C, I_0) = E_{DATA}(C, I_0) + E_{SHAPE}(C, C_T). \tag{0.3}$$

The term $E_{SHAPE}(C, C_T)$ measures the distance between $C$ and template shape $C_T$.

Functionals (0.2) resp. (0.3) are often quite complicated and require elaborate techniques for minimization. Common methods are graph theoretic approaches, gradient descent methods or stochastic algorithms, if gradients are difficult to compute. However, quite

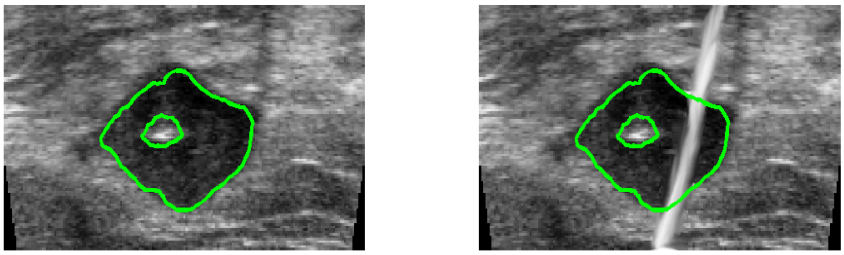


Figure 0.1: LEFT:A 2D slice of a 3D ultrasound image of a cyst and its segmentation. The hole inside the segmented object stems from a biopsy needle. RIGHT: Segmentation of an ultrasound image with a priori information. The ultrasound image has been corrupted by a bright stripe. If information on the shape of the cyst is available, approximately the same segmentation as on the left hand side should be obtained.

frequently, Euler-Lagrange equations of (0.2) and (0.3) can be computed, which in turn imply (discrete) iteration equations of type

$$C_0 = \text{initial start contour}$$

$$C_{n+1} = f(C_n, I_0) \text{ resp. } C_{n+1} = f(C_n, I_0, C_T), \tag{0.4}$$

such that $C_n$ converges to a minimizer of (0.2) resp. (0.3). It has turned out, that slight modifications of evolution equations of type (0.4) produce good segmentation results in the sense of the rough definition as well, although they cannot be derived from Euler-Lagrange equations of a functional (0.2) resp. (0.3). All these iterative models can be interpreted as a deformation of the initial contour $C_0$ towards the boundary of the object, and are therefore called deformable models.

In this dissertation, we study segmentation of topologically complex objects in 3D image data and provide new concepts and algorithms which improve existing approaches regarding efficiency and stability.
When segmenting objects of unknown topology with deformable models (0.4), the chosen initial contour $C_0$ is in general topologically different from the final segmenting contour, and therefore changes in topology during iteration are desired. While these changes happen automatically in implicitly formulated deformable models [89, 101] (which suffer from other shortcomings, like computational efficiency, artifacts in narrow band implementations), this is not the case in parametric deformable models [60], and a special topology adaptation system is required [84, 72, 17, 93].
We propose two main novel ideas to incorporate these topological transformations in

parametric deformable models.
The first idea is a two-step approach, published in [3]: In a first step, a pre segmentation with a deformable model (0.4) is carried out. During this evolution, topological transformations of the mesh are not performed, but information on contour self-intersections is obtained. In a second step, a topologically adapted segmentation is derived with a mesh construction algorithm from surface normals.
The second idea is developed in [2, 4]: A spatial hashing technique is used to detect self-collisions of the evolving discretized contour. For the topology adaptation, combinatorial formulas are derived from algebraic topology. The algorithm does not require global re-parameterizations and is computationally very efficient.

In segmentation of topologically complex objects with a priori knowledge (0.3), the efficiency and flexibility of the model crucially depends on the representation of the template shape $C_T$ and its corresponding shape space. As our main idea, we propose a shape space which combines advantages of medial axis shape spaces developed in [58, 91] and level set function shape spaces developed in [78]. Our shape space consists of medial ball representations, i.e. sets of balls which are located on the medial axis of an object. Unlike the medial axis shape space [58, 91], this shape space can be computed automatically from training data for arbitrary object topology. Furthermore, the medial ball shape space is computationally efficient, since it does not require computations on large grids as in [78]. We provide a statistical analysis of the medial ball shape space to label different ball representations, measure distances between ball representations and compute a mean ball representation. We employ medial ball shape spaces in segmentation with a simplified Mumford-Shah functional and demonstrate its advantages on synthetic and medical data.

## Outline

In Chapter 1, an introduction to segmentation via energy minimization methods and deformable models is given. Our focus is on topological adaptivity, flexibility and computational efficiency of the models. In Chapter 2, a two-step approach for topological adaptivity in parametric deformable models (0.4) is presented, which has already been published in [3].

In Chapters 3 and 4, combinatorial methods to incorporate topological adaptivity in deformable models (0.4) are developed. These ideas have already been published in [2] and [4]. Chapter 5 gives a topological flexible and computationally efficient algorithm for image segmentation with prior information (0.3) based on medial axis shape spaces. This part of the thesis has been submitted to ECAM - European Journal of Applied Mathematics.

Jochen Abhau Innsbruck, May 7, 2010

# Acknowledgement

First of all I want to thank Prof. Dr. Otmar Scherzer for his excellent guidance and scientific advice throughout the development of this thesis.

Furthermore, many thanks go to my colleagues at the Infmath Imaging Group, in particular Dipl.-Ing. Sebastian Colutto, for our fruitful discussions and good collaboration.

Moreover, I want to thank Dr. Bernhard Kornberger and Prof. Dr. Oswin Aichholzer from the Institute for Software Technology at the Graz University of Technology for the succesful teamwork over the last year.

I am grateful to Prof. Dr. Zakaria Belhachmi from the University of Metz for exchange of ideas and stimulating discussions.

Lust but not least, I am indebted to my parents, Helena and Jasper for their understanding, endless patience and encouragement.

My work has been supported by the Austrian Science Fund (FWF), project Y123-INF, project S10505-N20 and project 9203-N12 and the Industrial Mathematics Competence Center (IMCC), Linz.

# Contents

# 1 Background on energy minimization methods and deformable models for segmentation

In this chapter, we give an introduction to segmentation with energy minimization methods and active contours (also called deformable models). We focus on segmentation of topologically complex objects, which is the subject of this thesis.
We begin with a review of segmentation functionals in Section 1.1. The most common methods to minimize these segmentation functionals are active contour methods. In these models, the segmenting contour can be represented implicitly or parametrically. We review implicit active contours in Section 1.2, a review of parametric active contours is given in Section 1.3. In Section 1.4, we review graph theoretic approaches for segmentation, since they emerged quite recently as an alternative to active contour methods in minimization of segmentation functionals. Section 1.5 gives an introduction to segmentation with a priori information on the shape of the object to segment. We conclude this chapter with some material on mesh topology in Section 1.6, which is needed in Chapters 3 and 4 later on.

## 1.1 Segmentation functionals

In this section, we review common functionals of type (0.2) for segmentation of 2D and 3D objects in grey value images $I_0 : \Omega \to \mathbb{R}$, with $\Omega \subset \mathbb{R}^d$ and $d \in \{2, 3\}$. Let

$$\mathcal{C}^{2,m} = \{C : [0,1] \to \Omega \text{ piecewise m-times continuously differentiable with } C(0) = C(1)\} \tag{1.1}$$

and

$$\mathcal{C}^{3,m} = \{C : [0,1]^2 \to \Omega \text{ piecewise m-times continuously differentiable with } C(0,\cdot) = C(1,\cdot) \text{ and } C(\cdot,0) = C(\cdot,1)\}. \tag{1.2}$$

Functional (0.2) is constructed, such that minimizing elements are possible segmentations of the object of interest. In this functional, two terms are linked, a fit-to-data term and a smoothness term. The fit-to-data term detects the boundary of the object. In the literature, two main methods have been established for this purpose: Gradient-based segmentation and region-based segmentation.

*Gradient-based segmentation functionals*

If edges or corners appear in an image, the corresponding image intensity function has high gradients along these features. This observation is used in gradient-based segmentation models. These models were pioneered in [60], and are called snake models. Although snakes are originally introduced in 2D, they can be generalized to arbitrary dimensions $d$. Our representation follows the original two-dimensional setting in the literature.
For $\alpha, \beta, \lambda > 0$, minimizers of the functional

$$E(C) = \alpha \int_0^1 |C'(q)|^2 dq + \beta \int_0^1 |C''(q)|^2 dq - \lambda \int_0^1 |\nabla I_0(C(q))| dq. \tag{1.3}$$

over $C \in \mathcal{C}^{2,2}$ determine contours which bound segments. Here, the first two terms determine the smoothness of the curve (internal energy of the snake), and the last summand is small for curves which are contained in regions of high image gradient (fit-to-data term, external energy of the snake). For numerical implementation, an Euler scheme is used, implicit in the smoothness terms and explicit in the fit-to-data term.
In this model, the energy functional $E$ is not independent of the parametrization of a curve $C$. This point is overcome in the geodesic active contours model of [26]. They study the slightly simplified energy functional

$$E(C) = \alpha \int_0^1 |C'(q)|^2 dq - \lambda \int_0^1 |\nabla I_0(C(q))| dq \tag{1.4}$$

i.e., $\beta = 0$ in (1.3), arguing that minimzers of (1.4) are still sufficiently smooth. From (1.4), a similar segmentation functional is derived,

$$\min \int_0^1 g(|\nabla I_0(C(q))|)|C'(q)| dq \tag{1.5}$$

with a strictly decreasing function $g : [0, \infty) \to \mathbb{R}^+$. Functional (1.5) is independent of the parametrization of $C$ and its minimization can be regarded as computation of a geodesic in Riemannian space with metric $g$.

*Region-based segmentation functionals*

In contrast to the strategy of detecting edges, an alternative idea is to find image regions of homogeneous intensity. For this purpose, the following functional is minimized over $I \in C^1(\Omega, \mathbb{R})$ and $C \in \mathcal{C}^{d,1}$ in the work of [87]:

$$E(I, C) = \mu \mathcal{H}^{d-1}(C) + \lambda \int_\Omega (I_0(x) - I(x))^2 dx + \int_{\Omega \setminus C} |\nabla I(x)|^2 dx \tag{1.6}$$

In this functional, $\mathcal{H}^{d-1}(C)$ is the length ($d = 2$) or area ($d = 3$) of the segmenting contour, measured with the $(d-1)$-dimensional Hausdorff measure. The interesting idea incorporated in this segmentation functional is, that a smoothed image $I$ from $I_0$ and a segmenting contour $C$ are computed simultaneously. Looking closer at the terms involved in this functional, we see that the length (or area) of $C$ is small, if the contour

is expected to be smooth. The second term $\lambda \int_\Omega (|I_0(x) - I(x)|^2) dx$ is the fit-to-data term for the smoothed image $I$, and the last term $\int_{\Omega \setminus C} |\nabla I(x)|^2 dx$ imposes smoothness conditions on $I$ on regions inside and outside the contour.
A simplified version of (1.6) can be obtained, if the minimization is only performed over functions $I$ which are piecewise constant on connected components of $\Omega \setminus C$. Adding a further energy term which measures the enclosed area (resp. volume) of the contour, the Chan-Vese model [27] is obtained. For its formulation, let $\mathcal{I}(C) \subset \Omega$ be the inner part of $C$ and $\mathcal{O}(C) \subset \Omega$ the outer part of $C$. The functional to minimize is then given by

$$E(c_\mathcal{I}, c_\mathcal{O}, C) = \mu \mathcal{H}^{d-1}(C) + \nu \mathcal{H}^d(\mathcal{I}(C)) + \lambda_1 \int_{\mathcal{I}(C)} (I_0 - c_\mathcal{I})^2 \, dx + \lambda_2 \int_{\mathcal{O}(C)} (I_0 - c_\mathcal{O})^2 \, dx. \quad (1.7)$$

Here, the variables $c_\mathcal{I}$ and $c_\mathcal{O}$ are values of a piecewise constant image $I$, which is given by $c_\mathcal{I}$ on $\mathcal{I}(C)$ and $c_\mathcal{O}$ on $\mathcal{O}(C)$.

It is important to note, that Equations (1.4), (1.5), (1.6) and (1.7) do not depend on the parameterization of the contour $C$ and can as well be formulated *implicitly*. This means that $C$ is given as a $d-1$ dimensional submanifold of $\mathbb{R}^d$ by some defining equations.
The common framework to compute minimizers of Equations (1.5), (1.6) or (1.7) consists of computing Euler-Lagrange partial differential equations (PDE's). These PDE's are solved numerically with iterative schemes of type (0.4). For initialization, a contour is chosen which encloses the object and then shrinks, or which lies inside the object and grows. At each iteration step, the contour evolves closer to the boundary of the object to segment. Therefore, these methods are called *active contour methods* or *deformable models*. In the following two sections, we present some evolution schemes. Here we differ between *implicit* active contours, where the contour is represented implicitly, and *parametric* active contours, where the contour is parameterized as in (1.1) or (1.2).

## 1.2 Implicit active contours

The most common implicit active contour methods are *level set methods*, established in [89], and *fast marching methods*, established in [101]. We will discuss the principles of level set methods in Subsections 1.2.1 and 1.2.2 and efficient implementations [5, 115, 69] in Subsection 1.2.3 later on. Fast marching methods are reviewed in Subsection 1.2.4. Implicit methods share the common property, that topology changes of the contour happen automatically when evolving the contour. In many applications, this is a desirable feature. But under some circumstances explicit topological restrictions are imposed, such as connectedness of the final segmenting contour. Recently, topological restrictions have been implemented into level set evolutions and we discuss these works in Subsection 1.2.5 finally.

### 1.2.1 Level set methods

In level set implementations, a contour is represented as the zero level set of some real valued function. Let $C$ be a closed $(d-1)$-dimensional manifold, embedded in $\Omega \subset \mathbb{R}^d$.

A function

$$\phi : \Omega \to \mathbb{R} \tag{1.8}$$

is called a level set representation of $C$, if

$$\phi(x) = 0 \Longleftrightarrow x \in C. \tag{1.9}$$

Note that by the Jordan curve theorem [56, p.169], an *interior* $\mathrm{int}(C)$ and an *exterior* $\mathrm{ext}(C)$ of $C$ are well defined. The standard choice for $\phi$ is the signed distance function

$$\Omega \to \mathbb{R}, x \mapsto \begin{cases} -\min_{y \in C} \|x - y\| & \text{if } x \in \mathrm{int}(C) \\ 0 & x \in \mathcal{C} \\ \min_{y \in C} \|x - y\| & \text{if } x \in \mathrm{ext}(C). \end{cases} \tag{1.10}$$

In the original paper [89], the surface $C$, represented by $\phi$, is evolved over time in normal direction by the partial differential equation

$$\frac{\partial \phi}{\partial t} + F|\nabla \phi| = 0 \text{ on } \Omega \times [0, \infty). \tag{1.11}$$

Here, $F$ is an arbitrary scalar function, depending on the curvature of $C$. At time $t \geq 0$, the contour $C(t)$ is given by $\{x \in \mathbb{R}^d \mid \phi(x, t) = 0\}$. Note that as initial condition, $C(0) = C$ is required.

An appropriate solution concept for Equation (1.11) is given by *viscosity solutions* (for a complete exposition of this subject, see [34]), which allows for solutions which need not be differentiable.

### 1.2.2 Examples

For the following examples, note that the mean curvature $\kappa$ of the evolving level set function is given by

$$\kappa = \mathrm{div}\left(\frac{\nabla \phi}{|\nabla \phi|}\right) \tag{1.12}$$

In all examples, the initial value $\phi(x, 0)$ is restricted to be a fixed level set function $\phi_0$.

A motivating example for contour evolution (without taking image information into account) is mean curvature motion, which is obtained by setting $F(\kappa) = -\kappa$ in Equation (1.11).

$$\frac{\partial \phi}{\partial t} = |\nabla \phi| \ \mathrm{div}\left(\frac{\nabla \phi}{|\nabla \phi|}\right) \text{ on } \Omega \times [0, \infty). \tag{1.13}$$

The contour evolves in inward normal direction at mean curvature speed. As is shown in [52], the contour length (resp. area) is minimized by steepest descent by this evolution.

For segmentation of images, the image data itself has to be inserted in Equation (1.11),

thus stopping the evolution if the boundary of the object is reached.
In gradient based evolutions, edge indicator functions are used, such as

$$g = \frac{1}{1 + |\nabla I_0|} \tag{1.14}$$

and plugged into the following models.

(1) *Geometric active contours*
An evolution model, based on edge detection by high gradients, is given by the geometric active contour model [25]

$$\frac{\partial \phi}{\partial t} = g(x)|\nabla \phi| \left( \text{div} \left( \frac{\nabla \phi}{|\nabla \phi|} \right) + k \right) \text{ on } \Omega \times [0, \infty), \tag{1.15}$$

where $k$ is a real positive constant. As in mean curvature motion, the contour is evolved in direction of the inward normal, but at a speed depending on mean curvature, image data and $k$. As a consequence, the initial contour has to enclose the object to segment. The constant $k$ is chosen such that $\kappa + k$ is always positive, $k$ can be interpreted as a force pushing the contour towards the boundary of the object, when $\kappa$ becomes negative. If the contour moves over voxels with considerably different intensity values, $g(x) \approx 0$ and the contour tends to stop. On the other hand, the contour evolves over voxels of low contrast at some nonzero speed, which is approximately equal to $\kappa + k$ if $g$ is chosen as in (1.14).

(2) *Geodesic active contours*
The Euler-Lagrange equations for (1.5) in a level-set framework as derived in [26], give an evolution

$$\frac{\partial \phi}{\partial t} = |\nabla \phi| \left( \text{div} \left( g(x) \frac{\nabla \phi}{|\nabla \phi|} \right) + k g(x) \right) \text{ on } \Omega \times [0, \infty), \tag{1.16}$$

where $k$ is a real positive constant.
Similar to the geometric active contour model, the constant force $k$ pulls the contour towards the boundary of the object, which is detected by $g$.
Both the geometric and the geodesic active contour model are models, where the evolution of the contour $C$ is independent of its parametrization.

(3) *Chan-Vese model*
A level-set implementation of the Chan-Vese functional(1.7) is given by

$$\begin{aligned} \tfrac{\partial \phi}{\partial t} = \mu |\nabla \phi| \text{div} \left( \tfrac{\nabla \phi}{|\nabla \phi|} \right) - \nu - \lambda_1 (I_0 - c_{\mathcal{I}})^2 + \lambda_2 (I_0 - c_{\mathcal{O}})^2 \text{ on } \Omega \times [0, \infty) \\ \text{with} \qquad c_{\mathcal{I}} = \tfrac{1}{\mathcal{H}^d(\{x \mid \phi(x) < 0\})} \int_{\phi < 0} I_0(x) dx \\ \text{and} \qquad c_{\mathcal{O}} = \tfrac{1}{\mathcal{H}^d(\{x \mid \phi(x) > 0\})} \int_{\phi > 0} I_0(x) dx. \end{aligned} \tag{1.17}$$

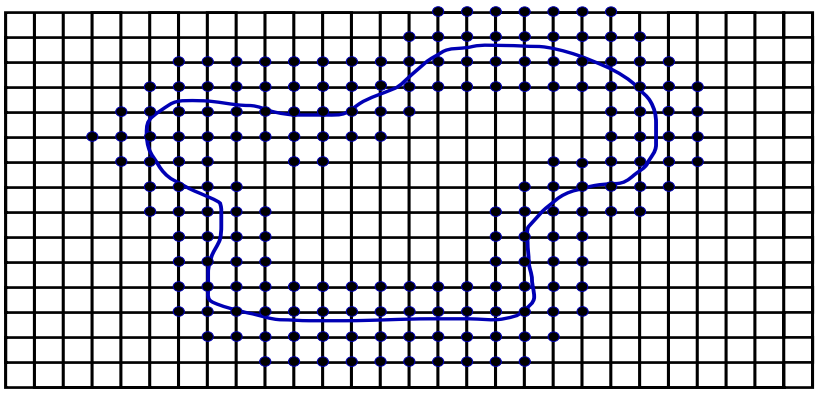


Figure 1.1: For evolution of the curve, only the values of $\phi$ on the dots (the narrow band) are computed

By this equation, evolution of $\phi$ is carried out on all level sets. Since the segmenting contour is implied by the zero level set of $\phi$, one is sometimes only interested in values of $\phi$ around its zero level line. In this case, one can replace $|\nabla\phi|$ by $\delta_\epsilon(\phi)$, a smooth approximation of the Dirac measure $\delta_0$ evaluated on $\phi$.
Arguing that the term $\mathcal{H}^{d-1}(C)$ in Equation (1.7) imposes sufficient regularity on the contour, one can set $\nu = 0$ and obtain a simpler version of Equation (1.17),

$$\frac{\partial\phi}{\partial t} = \mu\delta_\epsilon(\phi)\mathrm{div}\left(\frac{\nabla\phi}{|\nabla\phi|}\right) - \lambda\left((I_0 - c_{\mathcal{I}})^2 + (I_0 - c_{\mathcal{O}})^2\right) \text{ on } \Omega \times [0,\infty). \tag{1.18}$$

Thus, mean curvature motion is combined with iteratively updated terms, measuring the distance of the image intensity to computed mean intensities inside and outside of $C$.

### 1.2.3 Efficient implementations - the Narrow Band Method and Additive Operator Splitting

In this subsection, we are concerned with efficient numerical implementations of level set methods (1.11).

*Basic Numerical Implementation*

In order to solve (1.11) numerically, a grid of $N^d$ points is used to discretize $\Omega$, and on each point $(x_1, \ldots, x_d)$, the signed distance function $\phi_{x_1,\ldots,x_d}$ of the contour is stored and iteratively recomputed when solving (1.11). In the original work [89], an explicit upwind scheme is computed in each evolution step. All implementations of numerical schemes on the original grid are very costly, since each evolution step takes $\mathcal{O}(N^d)$ time, although the contour itself is only $(d-1)$-dimensional. Two main ideas have been

worked out to reduce computational costs when computing level set evolutions: *Narrow Band Methods* [5] and *Additive Operator Splitting* [69].

*Narrow Band Implementation*

Having a closer look at the discretization scheme above, we note that in fact only the values of $\phi$ on gridpoints close to the evolving contour are relevant for further evolution of the contour. Therefore, a (narrow) band is arranged around the zero level set of $\phi$ consisting of gridpoints close to the zero level set of $\phi$, and during an evolution, only the gridpoints lying in this band are processed. For an illustration, see Figure (1.1). Since the band reduces computational complexity by one dimension, resulting computational effort is $\mathcal{O}(n^{d-1})$. However, some technical difficulties have to be addressed when computing on narrow bands:

- After a number of iteration steps, the narrow band has to be updated, since the contour must not evolve out of the band. For this purpose, a scheme has to be implemented which detects, if the contour is still inside the narrow band.
- If the narrow band is too small, approximations of the partial derivatives of $\phi$ become inaccurate.

Therefore, implementing level set evolutions with narrow bands requires engineering.

*Additive Operator Splitting*

A fast semi-implicit scheme for level set evolutions is described in [114], focussing on efficient formulation of the evolution equations. The evolution equations (1.13), (1.16)and (1.15) all fit into the general framework

$$\frac{\partial \phi}{\partial t} = a(x)|\nabla\phi|\text{div}\left(\frac{b(x)}{|\nabla\phi|}\nabla\phi\right) + |\nabla\phi|kg(x) \text{ on } \Omega \times [0,\infty) \tag{1.19}$$

with real valued functions $a$ and $b$. A semi-implicit scheme is derived for solution of (1.19), which is of the form

$$\phi^{n+1} = \phi^n + \tau \sum_{l \in \{x_1,\ldots,x_d\}} A_l(\phi^n)\phi^{n+1}. \tag{1.20}$$

Here, $\tau$ is the time discretization parameter and $A_{x_i}(\phi^n)$ is a linear operator which maps $\phi^n_{x_1,\ldots,x_n}$ to a linear combination of elements $\phi_{x_1,\ldots,x_{i-1},y,x_{i+1},\ldots,x_n}$, with $y$ varying over neighbors in direction $x_i$. Therefore, each matrix $A_{x_i}(\phi^n)$ is tridiagonal and hence efficient to invert by Gauss algorithm. Equation (1.20) can be transformed into

$$\phi^{n+1} = (Id - \tau \sum_{l \in \{x_1,\ldots x_d\}} A_l(\phi^n))^{-1}\phi^n \tag{1.21}$$

In this expression, the sum of matrices $A_{x_i}(\phi^n)$ occurs, which is no longer tridiagonal and its inversion is therefore very time consuming. The important observation at this point is, that splitting the linear operators into

$$\phi^{n+1} = \frac{1}{2} \sum_{l \in \{x_1, \ldots x_d\}} (Id - 2\tau A_l(\phi^n))^{-1} \phi^n \tag{1.22}$$

has the same approximation order as system (1.21), but for its solution, only $d$ tridiagonal matrices have to be inverted. Therefore, system (1.22) can be solved very efficiently with the Gauss algorithm.

### 1.2.4 Fast marching methods

A related way to represent a closed submanifold $C \subset \mathbb{R}^d$ of codimension one implicitly is given by its *boundary value formulation*. In this representation, a function $T : \mathbb{R}^d \to [0, \infty)$ describes both the contour and its evolution, interpreting the value $T(x)$ as arrival time of the contour at point $x$. If $F$ denotes a positive valued real function, the evolution of $T$ is given by

$$|\nabla T| F = 1 \tag{1.23}$$

and the contour at time $t$ is given by $C(t) = \{x \in \mathbb{R}^d \mid T(x) = t\}$. Function $F$ can be interpreted as speed of the evolution, dependent on image features. There is an efficient numerical implementation of (1.23), based on the special structure of $T$. This scheme is comparable to the narrow band implementation of level set evolutions, and has even one advantage: Since $T$ measures distances of points from the contour, modification of the narrow band after an iteration is very efficient.
However, an important restriction of evolutions with fast marching methods is, that only outward evolution ($F > 0$) is allowed.

### 1.2.5 Preventing topology changes in implicit active contour evolutions

In several applications automated topology changes of the evolving contour are desirable. However, in some applications like biomedical image segmentation where the topology of the object to segment is known by anatomical knowledge, topology changes are not desirable. In this section, we review level set evolutions under topology control.

As one of the first works in this direction, [54] use concepts of digital topology to suppress topology changes during contour evolution in 3D. Assume that $x, y$ are 3-dimensional voxels. The following neighborhood relations can be defined between $x$ and $y$, compare [15]: If $x$ and $y$ share a face, they are called 6-connected, if they share an edge, they are 18-connected and if they share a corner, they are 26-connected. A binary image gives a decomposition of discrete voxel space $\Omega$ into a foreground component $X$, containing voxels of one intensity and a background component $\overline{X} = \Omega \backslash X$ containing voxels of the other intensity. The (local) neighborhood relationships defined above enable to define paths and hence (global) connectivity properties of foreground and background. This can be done consistently, when using $n$-connectedness for $X$, and $\overline{n}$-connectedness for $\overline{X}$, with

pairs $(n, \overline{n}) \in \{(6,26),(6,18),(18,6),(26,6)\}$. A point $x \in X$ is called *simple point*, if $X \backslash \{x\}$ has the same connectivity as $X$, and $\overline{X} \cup \{x\}$ has the same connectivity as $\overline{X}$. The interesting thing about simple points is, that they possess a *local* characterization: Point $x$ is simple, if the 26 voxels neighboring $x$ consist of exactly one foreground component (measured by $n$-connectivity) and exactly one background component (measured by $\overline{n}$-connectivity). In the topology controlled level set model of [54], voxels on which the level set function is positive belong to the foreground, the others to the background. During the evolution, the level set function is only allowed to change its sign on simple points. This basic scheme suppresses all topological changes, hence the final segmenting contour is homeomorphic to the initial contour. A generalization of this scheme is presented in [100], using *multisimple* points, which allow for topology control in further detail.

A different approach to control topology during level set evolutions has been suggested in [106] in 2D. There, the segmentation functional to minimize is supplemented by a functional which becomes large, if parts of the evolving curve are close. For a curve $C$, the energy

$$E_{2D,R}(C) = \frac{1}{2} \int \int_{C \times C} \left( \frac{1}{\|\mathbf{C}(s) - \mathbf{C}(\hat{s})\|} - \frac{1}{d_C(s,\hat{s})} \right) d\hat{s} ds \tag{1.24}$$

is defined, where $\|\cdot\|$ is the Euclidean norm in $\mathbb{R}^2$ and $d_C(s,\hat{s})$ is the geodesic distance between $\mathbf{C}(s)$ and $\mathbf{C}(\hat{s})$ along curve $C$. This energy becomes large, if the contour curve evolves close to self-intersection. Recently, this approach has been extended to 3D in [95].

Further work using penalty functionals to prevent topology changes in level set evolutions is provided in [10]. There, an interior curve $I_d = \{x + d\nabla\phi(x) \mid x \in \partial D\}$ and an exterior curve $E_l = \{x - l\nabla\phi(x) \mid x \in \partial D\}$ with distance parameters $d, l > 0$ to the evolving contour are maintained during level set evolution. The penalty functional is

$$H(\phi) = -\int_{\partial D} \log[\phi(x + d\nabla\phi(x))] ds - \int_{\partial D} \log[\phi(x - l\nabla\phi(x))] ds. \tag{1.25}$$

If two parts of the curve evolve closer than $\max\{d, l\}$, functional $H$ becomes large.

In the work of [76] it is observed that the signed distance function fulfils

$$\langle \nabla\phi(x), \nabla\phi(y) \rangle \approx -1 \tag{1.26}$$

at points where an evolving curve comes close to self-intersection and incorporate this term in a penalty functional prohibiting topology changes.

## 1.3 Parametric Implementation of Active Contours

In this section, we review parametric implementations of active contour models. The segmenting contour $C$ in $\Omega \subset \mathbb{R}^d$ is parameterized by a piecewise m-times continuously differentiable function

$$C : [0,1] \to \Omega \text{ such that } C(0) = C(1), \tag{1.27}$$

resp.

$$C : [0,1]^2 \to \Omega \text{ such that } C(0,\cdot) = C(1,\cdot) \text{ and } C(\cdot,0) = C(\cdot,1). \tag{1.28}$$

In most cases, $m = 1$ or $m = 2$. Implementations of gradient-based segmentation are outlined in Subsection 1.3.1, a survey on region-based segmentation is given in Subsection 1.3.2.

In standard curve ($d = 2$) or balloon ($d = 3$) evolutions, the contour is evolved over time starting from a contour $C_0$. Efficient parametric implementations are much simpler than implicit implementations, and produce accurate segmentations without artifacts, which can occur in implicit implementations with narrow bands. However, the main problem of parametric implementations is the lack of topological adaptivity. If $C_0$ parameterizes a simple closed curve or surface, every iterate is a simple closed curve as well, if no topological transformations are performed. In order to enable topological flexibility, various topology adaptation systems in parametric active contour evolutions have been considered in the literature. We will discuss these works in detail in Subsection 1.3.3.

### 1.3.1 Parametric implementations of gradient-based segmentation functionals

In segmentation functional (1.3), we set $E_{ext}(C) = \int_0^1 |\nabla I_0(C(q))dq|$. The term $E_{ext}(C)$ can be interpreted as external energy of the contour $C$. Functional (1.3) gives rise to the Euler-Lagrange equations

$$\frac{\partial}{\partial q}\left(\alpha\frac{\partial C(q)}{\partial q}\right) - \frac{\partial^2}{\partial q^2}\left(\beta\frac{\partial^2 C(q)}{\partial q^2}\right) - \nabla E_{ext}(C) = 0 \tag{1.29}$$

It is common to make the contour time dependent, such that iterative algorithms can be applied to solve (1.29).

$$m\frac{\partial^2 C(q,t)}{\partial t^2} + \gamma\frac{\partial C(q,t)}{\partial t} + \frac{\partial}{\partial q}\left(\alpha\frac{\partial C(q,t)}{\partial q}\right) - \frac{\partial^2}{\partial q^2}\left(\beta\frac{\partial^2 C(q,t)}{\partial q^2}\right) = \nabla E_{ext}(C) \tag{1.30}$$

Motion equation (1.30) is a particular version of motion by Newton's law. Interpreting $\frac{\partial^2}{\partial q^2}\left(\beta\frac{\partial^2 C(q,t)}{\partial q^2}\right) - \frac{\partial}{\partial q}\left(\alpha\frac{\partial C(q,t)}{\partial q}\right)$ as internal force $F_{int}$ acting on the evolving curve, and $\nabla E_{ext}(C))$ as external force $F_{ext}$, we obtain the following evolution by Newton's law:

$$m\frac{\partial^2 C}{\partial t^2} = -\gamma\frac{\partial C}{\partial t} + F_{int} + F_{ext} \tag{1.31}$$

The acceleration $\frac{\partial^2 C}{\partial t^2}$ of a mass point $C(t,q)$ of mass $m$ is equal to a sum of its velocity $\frac{\partial C}{\partial t}$ damped by $\gamma$ plus internal and external forces. The internal forces of

the model make the contour resist expansion or compression and make the model expand, while the external forces stop the snake when salient image features are reached.

### 1.3.2 Parametric implementations of region-based segmentation models

A parametric curve evolution for the Mumford-Shah functional (1.6), called *diffusion snake*, is presented in [36]. The curve $C$ is represented as a closed spline curve

$$C : [0,1] \to \Omega, C(q) = \sum_{n=1}^{N} p_n B_n(q) \tag{1.32}$$

where the $B_n$ are periodic, quadratic B-spline basis functions, and the $p_n$ are the spline control points. Since with this spline approach, the control points tend to overlap when measuring the curve length with the Hausdorff measure in (1.6), energy functional (1.6) is slightly changed into

$$E(I,C) = \mu \int_0^1 \frac{\partial^2 C}{\partial q^2} dq + \lambda \int_\Omega (I_0(x) - I(x))^2 dx + \int_{\Omega \backslash C} |\nabla I(x)|^2 dx. \tag{1.33}$$

Functional (1.32) is minimized alternating over contour $C$ and image $I$.

- For fixed image $I$, the Euler-Lagrange calculus gives

$$\frac{\partial E}{\partial C}[e^-(q) - e^+(q)]\ n(q) - \mu \frac{\partial^2 C}{\partial q^2} = 0 \text{ for } q \in [0,1] \tag{1.34}$$

  In this equation, $e^\pm = \lambda(I_0(C) - I(C))^2 \pm |\nabla I(C)|^2$ and $n$ is the outer normal on the contour. Solving (1.34) by gradient descent gives the evolution equation

$$\frac{\partial C}{\partial t} = -\frac{\partial E}{\partial C} = [e^+(q,t) - e^-(q,t)]n(q,t) + \mu \frac{\partial^2 C}{\partial q^2}(q,t) \text{ for } q \in [0,1], \tag{1.35}$$

  with a time parameter $t$ introduced. This equation is discretized and solved for the spline representation of the curve.

- For a fixed curve $C$, Euler-Lagrange equations are computed for image variable $I$ and the diffusion process

$$\frac{\partial I}{\partial t} = \text{div}(w_c \nabla I) + \lambda(I_0 - I) \tag{1.36}$$

  is obtained, $w_c$ being the characteristic function of the set $\Omega \backslash C$.

### 1.3.3 Topology adaptive implementations of parametric active contours

In numerical implementations of (1.31), the evolved contour $C$ is mostly discretized by polygons or meshes. This way, normals, curvature etc. are approximated by discrete mesh operators. However, the evolved contour cannot change its topology type during evolution, unless special topology adaptation systems are implemented. In this subsection, we study such systems, with a focus on 3D implementations.

*Topology adaptations in mesh-like structures.*
One of the first works which models surface evolutions allowing for topological transformations is [108]. There, particles are evolved, which carry similar information as the vertices of a mesh like position, normal vector and orientation, but do not contain any neighborhood relations. In order to obtain a surface mesh after an evolution step, a Delaunay triangulation of the particles is computed.

In [37, 38], the evolving surface is discretized as a 2-simplex-mesh, which is defined as a mesh where every vertex is adjacent to three edges. Topological complex meshes are put together from tubes or spheres for initialization and then evolved towards object boundary.

*The Delaunay approach of [93].*
The topology adaptive model of [93] relies on iterative remeshing with the restricted Delaunay triangulation. We first explain the geometrical concepts involved: For a set of points $E = \{p_1, \dots, p_n\} \subset \mathbb{R}^d$ and $\Omega \subset \mathbb{R}^d$, the restricted Delaunay triangulation $\mathrm{Del}_{|\Omega}(E)$ is defined as a subcomplex of the Delaunay triangulation $\mathrm{Del}(E)$. It consists of simplices of $\mathrm{Del}(E)$, whose dual Voronoi cells have nonempty intersection with $\Omega$. For example, in dimension $d = 3$, $\mathrm{Del}_{|\Omega}(E)$ is an approximation of $\Omega$ with tetrahedrons. Under the assumption that $E$ is a sufficiently dense sample of $\Omega$ and $\Omega$ is smooth, $\mathrm{Del}_{|\Omega}(E)$ is an approximation of $\Omega$. This tetrahedrization can be computed in $\mathcal{O}(|E| \log |E|)$ time (in case $d = 3$). Using the restricted Delaunay triangulation $\mathrm{Del}_{|\Omega}(E)$, a topology adaptive contour evolution step is performed in the following way:

At iteration step $n$, a point set $E_n$ (representing samples of the evolving contour), set $\Omega_n$ (representing the space enclosed by the contour) and $D_n = \mathrm{Del}_{|\Omega_n}(E_n)$ are given. The points $E_n$ are evolved to $E_{n+1}$ by evaluating Equation (1.31). Here, the structure of $D_n$ is taken into account in order to compute discrete derivatives. After this evolution, $D_n$ is no valid triangulation of the evolved point set $E_{n+1}$, since overlappings can occur. Therefore, the set $\Omega_{n+1}$ covered by $E_{n+1}$ with the (old) triangulation $D_n$ is computed, and the restricted Delaunay triangulation $D_{n+1} = \mathrm{Del}_{|\Omega_{n+1}}(E_{n+1})$ is calculated with respect to $\Omega_{n+1}$. After this update step, $D_{n+1}$ is a valid triangulation of the point set $E_{n+1}$, and $\Omega_{n+1}$ is the updated segmented object. By this approach, the topology of the evolving contour is allowed to vary between different iterations.

*The fast-marching-methods-like approach of [17].*
In [17], a discrete snake model is presented in 2D. Euclidean space $\mathbb{R}^2$ is supplemented with the grid $\mathbb{Z}\times\mathbb{R}\cup\mathbb{R}\times\mathbb{Z}$. A discrete snake, called r(estricted)-snake there and consisting of vertices and edges assembled to a polygon, is evolved under certain restrictions:

- the r-snake vertices lie on the grid,
- if an r-snake vertex reaches a gridpoint, i.e. an intersection point of grid lines, it is split,
- the r-snake is only evolved in normal direction.

During evolution, the computed normals are projected on the grid, and the vertices of the r-snake are translated by these normals. Time stepping $\Delta t$ is chosen, such that a vertex cannot move farther than the next gridpoint:

$$\Delta t = \min_V\{(1 - d_V)/v_V\} \tag{1.37}$$

In this equation, $d_V \in [0, 1]$ is the distance of vertex $V$ to the next gridpoint, and $v_V$ the evolution velocity of $V$ along the unit normal vector. This rather slow evolution speed enables unambiguous self-intersection detection and topology adaptation by local procedures on each square. This model has some similarity to the fast marching method in implicit active contour evolutions, see Subsection 1.2.4, since its evolution is parameterization independent and main information on the evolving contour is stored in the distance variable $d_V$.

*Works of Lachaud, [70], [72], [73],[71].*
In these works, a triangular mesh is evolved over time. In order to have some guarantees on the mesh at any time of the evolution, a parameter $\delta > 0$ is associated to the evolving mesh. It determines two geometric constraints:

- every edge fulfils $\delta \leq$ edge length $\leq 2\frac{1}{2}\delta$,
- two non-neighboring vertices cannot come close, $\|u - v\| \geq \frac{2\frac{1}{2}}{\sqrt{3}}\delta$.

After each evolution of the mesh vertices by Equation(1.31), a distance field is computed to check these conditions and perform local mesh refinement or coarsening if necessary. Self-collisions are detected, if the contour violates the second geometric constraint. This basic geometric topology adaptation framework has been applied to a Riemannian space setting, where distances are measured depending on the properties of the object to segment, thus allowing for a resolution adaptive implementation of the mesh.

*Topology adaptive snakes [83] of and [84].*
In these works, a Freudenthal triangulation of the image domain is utilized. This triangulation is constructed by subdividing the $d$-dimensional image domain into a uniform cubic grid and further subdividing each cube into $d$ factorial simplices. After an evolution step of the contour by Equation (1.31), the contour is reparameterized such that every vertex lies on an edge of one of the Freudenthal triangles. Locally on each Freudenthal triangle, self-intersections of the contour are detected and topological transformations are performed, which is similar to [17]. The evolution speed has to be chosen sufficiently small, such that no ambiguities occur in this framework.

## 1.4 Graph theoretic segmentation approaches

In graph theoretic segmentation approaches, discrete images are regarded as weighted graphs, and theory as well as algorithms from combinatorial optimization are transferred to image segmentation problems. In this section, we review popular methods for image segmentation via graph theory. These include rather direct ones like [45, 102] in Subsection 1.4.1 and also methods for minimization of segmentation functionals like (1.7) by graph cuts (Subsection 1.4.4). Theory on graph cuts is presented in the intermediate Subsections 1.4.2 and 1.4.3.

In order to interpret (discrete) images as graphs, let $\mathcal{P}$ be the set of pixels, corresponding to vertices in graph interpretation. The edge set $\mathcal{N}$ of the graph is given by two neighboring pixels. The neighborhood relation can be understood in several ways, for example 4- or 8-neighborhood in 2D, 6-, 18- or 26- neighborhood in 3D. In some cases, complete graphs are considered as well, i.e. each two vertices are connected by an edge. An image intensity function $I_0$ induces weights on the graph edges by setting

$$w_{p,q} = |I_0(p) - I_0(q)| \text{ for neighboring pixels } p, q. \tag{1.38}$$

We first review greedy segmentation algorithms working directly on the graph structure.

### 1.4.1 Greedy algorithms

From a decomposition of an image into different segments, one can in principle expect that two pixels of highly differing intensity should lie in different segments, whereas pixels of similar intensity lie in the same segment. This observation can be translated directly into a segmentation algorithm of greedy type. The image is considered as a weighted graph as in (1.38), and as initialization of an iterative procedure, every vertex (=pixel or voxel) is considered as a different segment of the image. The edges are sorted by weights in non-decreasing order, and iteratively, in a greedy manner, two segments are merged if they are connected by an edge with currently minimal weight. This algorithm is based on purely local criteria and fails to work on most real world images. As an improvement on the basic greedy algorithm, the Felzenszwalb-Huttenlocher algorithm [45] applies convolution with a Gaussian kernel to $I_0$ in a preprocessing step, thus reducing the influence of noise. Then the basic greedy algorithm is supplemented

by a criterion, which actually decides if two components are merged. Here, internal differences, i.e. the largest weight in the minimal spanning tree of a component, and differences between components, i.e. the smallest weight of an edge connecting the two components, are taken into account. By this criterion, global guarantees on the segmentation result can be inferred.

Note that after algorithms of greedy type, different segments are basically separated by edges of high weights. In noisy images, this strategy sometimes fails to produce good results, since single edges have a high impact on the segmentation. A more stable concept consists of computing segments, such that the sum of weights between different segments is maximized. Thus, the influence of single edges is reduced and more data variability inside a segment is enabled. This class of algorithms can still be solved efficiently, using the *minimal cut - maximal flow* duality in graph theory.

### 1.4.2 Background from graph theory - maximal flow and minimal graph cuts

Assume $G = (\mathcal{P}, \mathcal{N})$ is a directed graph with nonnegative edge weights $w_e$ for $e \in \mathcal{N}$, as can occur in our image segmentation context. Furthermore, let $s, t \in \mathcal{P}$ be two distinct vertices, called source and sink. Interpreting the edge weights as maximal capacities (i.e. maximum amount of flow which can pass), a flow is defined as a mapping $f : \mathcal{N} \to \mathbb{R}^+$ such that the capacity condition

$$f_e \leq w_e \text{ for all } e \in \mathcal{N} \tag{1.39}$$

and the flow conservation condition

$$\sum_{(q,p)\in\mathcal{N}} f_{(q,p)} = \sum_{(p,q)\in\mathcal{N}} f_{(p,q)} \text{ for all } p \in \mathcal{P}\backslash\{s,t\} \tag{1.40}$$

hold. The flow conservation condition guarantees that the amount of flow going into a vertex equals the amount flowing out again. The maximum flow problem consists of computing a flow $f$ which maximizes

$$|f| = \sum_{e\in\mathcal{N}} f_e. \tag{1.41}$$

On the other hand, an s-t cut of $G$ is a partition of $\mathcal{P}$ into disjoint subsets $S, T$ such that $s \in S$ and $t \in T$. The minimum cut problem consists of computing an s-t cut $\mathcal{P} = S \cup T$ minimizing

$$cut(S,T) = \sum_{p\in S, q\in T, (p,q)\in\mathcal{N}} w_{p,q} \tag{1.42}$$

As shown in [6], the minimal cut problem is dual to the maximum flow problem. For the max-flow problem with integer edge weights, efficient algorithms like the Ford-Fulkerson algorithm [6] exist, which solve the maximum flow problem in $O(|\mathcal{P}|^2|\mathcal{N}|)$ time. Hence by duality, (1.42) can as well be solved efficiently by the Ford-Fulkerson algorithm.

### 1.4.3 Segmentation algorithms based on minimal graph cuts

The theory above is applied to segmentation by replacing $w_e$ by $\max_{\tilde{e}\in\mathcal{N}} w_{\tilde{e}} - w_e$, thus transforming a maximization problem into a minimization problem. As with greedy segmentation algorithms, difficulties arise when directly computing minimal cuts of an image represented as in (1.38). This is for example done in [117]. Since the cut defined in (1.42) increases with the number of edges connecting two segments, the direct graph cut strategy would favor cutting small sets of isolated vertices in the graph. Notable improvements have been made in the normalized cuts algorithm [102]. This algorithm does not search for segments minimizing (1.42), but minimizes instead

$$Ncut(A,B) = \frac{cut(A,B)}{cut(A,\mathcal{P})} + \frac{cut(A,B)}{cut(B,\mathcal{P})} \tag{1.43}$$

Here, $cut(A,\mathcal{P})$ and $cut(A,\mathcal{P})$ serve as normalizing quantities and as consequence, small segments $A$ do not necessarily have a small normalized cut value $Ncut(A,\mathcal{P}\backslash A)$ and are thus prevented. Unfortunately, minimization problem (1.43) can no longer be minimized efficiently. Therefore, (1.43) is formulated for real weights and an approximate solution is proposed in [102].

More general, the graph cuts technique can be applied to minimization of segmentation functionals like (1.7). In the following, we explain this general theory.

### 1.4.4 General energy minimization via minimal graph cuts and applications

In this part, we explain how general energy minimization problems can be interpreted as graph cut problems and discuss its applications to image segmentation.
Consider a finite set $F$, an ordered set $\mathcal{P}$ and a functional

$$E : F^{\mathcal{P}} \to \mathbb{N} \tag{1.44}$$

which can be written in the form

$$E(I) = \sum_{p\in\mathcal{P}} E_p(I_p) + \sum_{p,q\in\mathcal{P},p<q} E_{p,q}(I_p, I_q) \tag{1.45}$$

with functionals $E_p : F \to \mathbb{N}$, $E_{p,q} : F^2 \to \mathbb{N}$.
In the image processing context, $\mathcal{P}$ is the finite set of pixels or voxels, and $F$ is the set of gray values an image $I$ can attain, typically $F = \{0,\dots,255\}$.
The graph theoretic approach to minimize (1.45) consists of constructing a graph $\tilde{G} = (\tilde{\mathcal{P}},\tilde{\mathcal{N}})$, such that minimizers of (1.45) can be computed from minimal cuts of $\tilde{G}$. A general solution theory has been developed in [67] for sets $F = \{0,1\}$. We repeat its main elements here, since it gives a criterion which functionals can be minimized via graph cuts and a constructive proof. The main result states, that (1.45) can be minimized by graph cuts, if and only if

$$E_{p,q}(0,0) + E_{p,q}(1,1) \leq E_{p,q}(0,1) + E_{p,q}(1,0) \text{ for all } p,q \in \mathcal{P} \tag{1.46}$$

holds. The graph $\tilde{G}$ is constructed in the following way:

1. Define additional vertices $s$ and $t$ as source and sink, and set $\tilde{\mathcal{P}} = \mathcal{P} \cup \{s, t\}$.
2. if $E_p(0) < E_p(1)$, then add the edge $(s, p)$ with weight $E_p(1) - E_p(0)$ to the graph, otherwise add $(p, t)$ with weight $E_p(0) - E_p(1)$ to the graph (for all $p \in \mathcal{P}$).
3. set $A = E_{i,j}(0,0)$, $B = E_{i,j}(0,1)$, $C = E_{i,j}(1,0)$ and $D = E_{i,j}(1,1)$. Note that

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} A & A \\ A & A \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ C-A & C-A \end{bmatrix} + \begin{bmatrix} 0 & D-C \\ 0 & D-C \end{bmatrix} + \begin{bmatrix} 0 & B+C-A-D \\ 0 & 0 \end{bmatrix} \tag{1.47}$$

   In the sum on the right hand side, the first term is constant, the second term only depends on the value of $I_p$ and the third term only depends on the value of $I_q$. Therefore these terms can be treated as in item (1). Concerning the last term, an edge $(v_i, v_j)$ with weight $E_{p,q}(0,0) + E_{p,q}(1,1) - E_{p,q}(0,1) - E_{p,q}(1,0)$ is inserted into graph $\tilde{G}$, which is greater or equal to zero because of the assumption.

Since all weights in this graph are greater or equal to zero, a minimal cut of this graph is well-defined and can be computed efficiently by the Ford-Fulkerson algorithm. As can be seen easily, a minimal s-t cut of this graph induces a binary valued function $I$ which is a minimizer of (1.45).

In the following, we describe the graph cut algorithm for minimization of the Chan-Vese functional (1.7) with $\nu = 0$,

$$E(c_{\mathcal{I}}, c_{\mathcal{O}}, C) = \mu \mathcal{H}^{d-1}(C) + \lambda_1 \int_{\mathcal{I}(C)} (I_0 - c_{\mathcal{I}})^2 \, dx + \lambda_2 \int_{\mathcal{O}(C)} (I_0 - c_{\mathcal{O}})^2 \, dx. \tag{1.48}$$

For minimization via graph cuts, this functional (1.48) is discretized in [120], see also [43], in the following way:

- $\Omega$ is discretized by a regular grid of pixels (or voxels) $p \in \mathcal{P}$.
- regions implied by the contour $C$ are translated into values $I_p \in \{0, 1\}$ for $p \in \mathcal{P}$, by the rule

$$I_p = \begin{cases} 1 & \text{if } p \in \mathcal{I}(C) \\ 0 & \text{if } p \in \mathcal{O}(C). \end{cases} \tag{1.49}$$

- the curve length (resp. surface area) of $C$ is approximated using a discrete Cauchy-Crofton formula. The Cauchy-Crofton formula enables to compute the curve length (resp. surface area) of $C$ using the number of intersections with given lines. The exact derivation of formulas in 2D and 3D can be found in [21]. As system of lines $e_1, \ldots, e_k$ and weights $\tilde{w}_1, \ldots, \tilde{w}_k$ (both independent of $C$) is obtained, such that

$$\mathcal{H}^{d-1}(C) \approx \sum_{k=1}^{n} n_C(k) \tilde{w}_k \tag{1.50}$$

The term $n_C(k)$ denotes the number of intersections of $C$ with $e_k$. For a contour $C$ implicitly given by a binary image $I$, it can written in the form

$$n_C(k) = \sum_{(p,q)\in\mathcal{P}^2,(p,q)\cap e_k\neq\emptyset} (1-I_p)I_q + I_p(1-I_q) \tag{1.51}$$

Altogether, we obtain

$$E(c_{\mathcal{I}}, c_{\mathcal{O}}, I) \approx \sum_{p\in\mathcal{P}} \left( \mu \sum_{k=1}^{n} \sum_{q\in\mathcal{P},(p,q)\cap e_k\neq\emptyset} \tilde{w}_k[(1-I_p)I_q + I_p(1-I_q)] \right) + \lambda_1(I_{0,p}-c_{\mathcal{I}})^2 I_p + \lambda_2(I_{1,p}-c_{\mathcal{O}})^2(1-I_p) \tag{1.52}$$

To a fixed binary image $I$, the optimal value of $c_{\mathcal{I}}$ and $c_{\mathcal{O}}$ in (1.52) is given by

$$c_{\mathcal{I}} = \frac{\sum_{p\in\mathcal{P}} I_p I_{0,p}}{\sum_{p\in\mathcal{P}} I_p} \tag{1.53}$$

and

$$c_{\mathcal{O}} = \frac{\sum_{p\in\mathcal{P}} (1-I_p) I_{0,p}}{\sum_{p\in\mathcal{P}} (1-I_p)}. \tag{1.54}$$

Functional (1.52) is minimized by an alternating procedure: After choosing an arbitrary initial contour $C$ (and hence $I$),

1. values of $c_{\mathcal{I}}$ and $c_{\mathcal{O}}$ from the value of $I$ are computed by (1.53) and (1.54).
2. for fixed values of $c_{\mathcal{I}}$ and $c_{\mathcal{O}}$, functional (1.52) is a special case of the general functional (1.45) and minimized by graph cuts, and goto (1).

The iteration is stopped if the value of (1.52) remains constant.

There are two main advantages of minimizing the Chan-Vese functional (1.7) by graph cuts:

- the result of the iterative optimization procedure is independent of the initialization.
- the graph cut algorithm always finds a global extremum.

## 1.5 Segmentation with prior information

The segmentation models presented in Sections 1.1, 1.2 and 1.3 work well in situations, when the image data is uncorrupted, as has already been pointed out in the Preface. But if noise, low contrasts, partial occlusions or weak object boundaries are present, active contour models without a priori information on the object to segment tend to fail. Therefore, it is useful to incorporate a shape prior (if available) into the segmentation model.

To do that, either uncorrupted training images are segmented with methods of the previous sections, or corrupted data is segmented by an expert and the segmented objects are used to compute a statistical shape model. Standard statistical concepts utilized herein are *principal component analysis (PCA)* and *Mahalanobis distance*. These concepts are reviewed in Section 5 and we omit a discussion in this section. Furthermore, there are several ways to represent the shape prior and define shape space, and we list the most common ones in Subsection 1.5.1. The shape prior is then inserted in the active contour model, mainly by a Bayesian approach or by adding a further penalty term (shape energy) to the energy functional, see Subsection 1.5.2. Finally, we review some recent work on segmentation with a priori shape knowledge in Subsection 1.5.3.

### 1.5.1 Shape spaces and representations of the shape priors

A priori knowledge on the object to segment is often knowledge on size or shape. We define the shape of an object as collection of data of an object, which is invariant under similarity transformations, i.e. translations, rotations and scaling as in [62]. In the segmentation literature, shapes are represented in several different ways, and we give some shape models here.

*Medial axis representation*
A shape representation which has been introduced in [49, 92] and studied extensively [58, 91, 80, 47, 46, 48] is the *Medial Axis Representation* (M-Rep) of shapes. Generally, the medial axis of an arbitrary open set $X$ in Euclidean space is defined as the set of points which have at least two closest points in the complement of $X$. (For an extensive survey on medial axes computation, see [14]). The medial axis is discretized by a mesh consisting of *medial atoms* and edges connecting the atoms. Medial atoms carry the following data:

- a position $x \in \mathbb{R}^3$
- a distance $r > 0$ to two or more implied boundary positions
- a local coordinate frame $F \in SO(3)$, parametrized by three orthonormal vectors $(n, b, b^{\perp})$, where $n$ denotes the normal to the medial mesh and $b$ is a tangent vector of the implied boundary, pointing in direction of steepest implied volume descent.
- an object angle $\theta \in [0, \frac{\pi}{2}]$, which is the angle between the implied boundary and $b$.

Medial atoms which lie on the boundary of the mesh carry a parameter $\eta$ as extra data. For a visualization of medial atoms, see Figure 1.2. The implied boundary of an object represented by an M-rep is reconstructed by Bezier spline interpolation. A distance measure between M-reps is obtained by regarding M-reps (consisting of $n_1$ inner and $n_2$ boundary medial atoms) as elements of the Riemannian manifold $(\mathbb{R}^3 \times \mathbb{R}^+ \times \mathrm{SO}(3) \times [0, \frac{\pi}{2}])^{n_1} \times (\mathbb{R}^3 \times \mathbb{R}^+ \times \mathrm{SO}(3) \times [0, \frac{\pi}{2}] \times \mathbb{R}^+)^{n_2}$.

The main advantage of M-reps is its stability under small perturbations of the implied boundary. Furthermore, meshes of different scale can be combined, therefore allowing

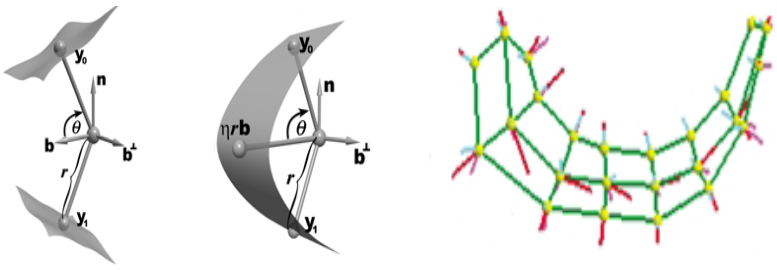



Figure 1.2: Medial Atoms (taken from [91]). On the left hand side, an inner medial atom, in the middle, a boundary medial atom is shown. The points $y_0$ and $y_1$ lie on the surface. On the right hand side, a mesh of medial atoms is shown.

for a multiresolution representation of objects by M-reps. On the other hand, there is no simple method to extract a medial atom mesh from segmented objects automatically, and some manual calibration is required.

*Sampling of boundary points*
Alternatively, the shape models can base on sample points. A representation employed in [33] consists of a uniform sampling of the object boundary. In order to cancel out the influence of similarity transformations, distances between point samples are measured by the Procrustes distance.
A further point based approach is the landmark approach detailed in [20], where instead of a uniform sampling of the object boundary, points are only placed at salient boundary features.

*Level set shape representation*
A shape representation which is suitable for active contours in the level-set formulation has been introduced in [78]. In the basic numerical implementation of level sets (Subsection 1.2.3) the signed distance function is sampled on a grid at $N^d$ points. The implied discrete surface can therefore be considered as a point in high dimensional space $\mathbb{R}^{N^d}$. The shape of an object is obtained by taking equivalence classes modulo similarity transformations in $\mathbb{R}^{N^d}$.

### 1.5.2 Bayesian models versus variational models

A priori shape knowledge is incorporated into segmentation functionals, by penalizing deviations of the shape of the segmenting contour to the precomputed mean shape. From a conceptual point of view, these penalizations can be carried out in two ways, a Bayesian framework and a variational framework.

*Bayesian framework*

In a Bayesian framework, shape priors are incorporated into the segmentation algorithm via *maximum a posteriori*(MAP) estimators. For related applications of MAP estimators in imaging, we refer to [99].Assume that $f(C|\theta)$ is the conditional probability density function of some probability distribution, which is dependent on parameter $\theta$. Furthermore, assume that $h(\theta)$ is a density function for the distribution of $\theta$. Then Bayes theorem for probability densities states that

$$h(\theta|C) = \frac{f(x|\theta)h(\theta)}{\int f(x|\tilde{\theta})h(\tilde{\theta})d\tilde{\theta}}. \tag{1.55}$$

Maximizing $h(\theta|x)$ for $\theta$ leads to the MAP estimator

$$\hat{\theta}_{MAP}(x) = \text{argmax}_\theta h(\theta|x) = \text{argmax}_\theta f(x|\theta)h(\theta) \tag{1.56}$$

In this equation, the normalisation term in the denominator is omitted since it is independent of $\theta$ and therefore irrelevant in maximization. The MAP estimator $\hat{\theta}_{MAP}(x)$ can be interpreted as parameter $\theta$, for which the probability to observe $x$ is maximal under the assumption that $x$ depends on $\theta$ and assuming a certain distribution on parameter $\theta$.

In the segmentation context, we are given training shapes depending on some parameter $\theta$. Let $\mu$ be the computed mean shape and $d_{MAHAL}$ be the Mahalanobis distance between shapes. It is common to define a probability density function from these data on shapes (for example in [78]) by

$$h(\theta) = D\exp(-d_{MAHAL}(\theta, \mu)). \tag{1.57}$$

Here, $D$ is a positive normalizing constant. This function is maximal for $\theta = \mu$, otherwise it becomes small.
Interpreting $x$ as the segmenting contour, and plugging a density function as (1.57) into (1.56), one can estimate the most probable shape $\theta$ when observing $x$, knowing a priori that $\mu$ is the mean shape, and which variability among shapes is present.
In Subsection 1.5.3, we give some examples in the literature using such an approach for segmentation with a priori information.

*Variational framework*
In a variational framework, shape priors are entered into segmentation functionals as a *shape energy*. Typically, a segmentation functional of type

$$E(C) = E_{DATA}(C) + \lambda E_{SHAPE}(C, C_T) \tag{1.58}$$

is minimized. Here, $E_{DATA}(C)$ can be for example functional (1.3) or (1.4), and $E_{SHAPE}(C)$ is a functional penalizing deviations of $C$ from a priori expected contours. We give some examples using this approach in Subsection 1.5.3.

### 1.5.3 Recent Work on Segmentation with Prior Information

In this subsection, we present recent work in segmentation with prior information utilizing the concepts discussed in Subsections 1.5.1 and 1.5.2.

In [78], a level set approach is developed to implement the geodesic active contour model 1.16. The a priori shape is represented in the level set representation of Subsection 1.5.1. This shape model is integrated into the geodesic active contour evolution as MAP estimator of the shape (1.56). The (discrete) evolution equation reads as

$$\phi^{t+1} = \phi^t + \lambda_1 v(t) + \lambda_2 (r(\hat{\theta}_{MAP}(\phi^t) - \phi(t)). \tag{1.59}$$

Here, $r$ is a map taking a shape to its implied segmenting contour and $v(t)$ is the geodesic active contour evolution speed, depending on image data $I_0$ and curvature of the contour. Starting from this work, several modifications and improvements have been made in the last years. An evolution similar to [78] has been studied in [113]. The only difference is, that the Chan-Vese functional (1.17) is employed instead of the geodesic active contour model.

In the segmentation model of [30], the shape model is implemented in the variational formulation (1.58), compare also Equation (1.5).

$$E(C, f) = \int_0^1 g(|\nabla I_0(C(q))|)|C'(q)|dq + \frac{\lambda}{2} d^2(f(C), C_T)|C'(q)|dq \tag{1.60}$$

In this equation, $f$ is a similarity transform, aligning template shape $C_T$ to $C$, and $d$ is a distance measure. For minimization of (1.60) over $C$ and $f$, a level set evolution is derived.

The approach chosen in [36] also fits into the general variational framework (1.58), but its implementational details differ from the concepts above in several ways. The segmenting contour $C$ is represented parametrically as a spline. The term $E_{DATA}(C)$ is given the Mumford-Shah functional (1.6), the Hausdorff measure of the contour length (resp. area) being replaced by a squared $L_2$ norm. As shape energy $E_{SHAPE}(C)$, the Mahalanobis distance (5.20) is utilized.

More recently, in [23] gradient based and region based segmentation is combined in a level set framework with shape priors. The proposed energy functional is

$$E(C, C_T, f) = \beta_s E_{SHAPE}(C, C_T, f) + \beta_b E_{BOUNDARY}(C, I_0) + \beta_r E_{REGION}(C_T, f, I_0) \tag{1.61}$$

In this functional, $E_{SHAPE}$ is given by the distance of $C$ from template shape $C_T$ (aligned by $f$) measured by distances of coefficients in a reduced basis after PCA, $E_{BOUNDARY}$ is defined as in Equation (1.5) and functional $E_{REGION}$ is similar to functional (1.7) without regularization, i.e. $\mu = \nu = 0$ in (1.7).

Attempts to generalize these ideas were made in [51].

The medial axis representation of the prior shape is implemented in a segmentation functional of type (1.58) in [91]. Recent applications of M-reps with Mumford-Shah

segmentation are presented in [31], using a CMA algorithm for minimization of a Mumford-Shah energy over the Riemannian shape manifold.

## 1.6 Background on Topology and Homology

In this section, we give some background on topology and homology theory, which is required for a better understanding of Chapters 3 and 4. We start with basic definitions, and then list some theorems which enable computation of homology groups for various spaces. For bibliographic sources, we refer to [96, 82, 56], background on algebra can be found in [75].

### 1.6.1 Basic definitions

Homology groups can be most easily defined for topological spaces which are built from simple objects, namely simplices.

**1.6.1 Definition.** Let $\{v_0, \dots, v_n\}$ be an affine independent subset of some Euclidean space. An $n$-simplex $[v_0, \dots, v_n]$ is the convex hull of $\{v_0, \dots, v_n\}$. The points $v_0, \dots, v_n$ are called vertices of the simplex, and every simplex whose vertex set is a subset of $\{v_0, \dots, v_n\}$ is called face of $[v_0, \dots, v_n]$. The number $n$ is called the dimension of $[v_0, \dots, v_n]$.

If a topological space consists of simplices, it is called a simplicial complex.

**1.6.2 Definition.** A finite simplicial complex $K$ is a finite set of simplices, such that

1. if $s \in K$, then each face $s'$ of $s$ is also an element of $K$.
2. if $s, t \in K$, then $s \cap t$ is either empty or a face of both $s$ and $t$.

The geometric realization $|K|$ of a simplicial complex $K$ is defined as the union of all its simplices,

$$|K| = \bigcup_{s \in K} s. \tag{1.62}$$

Reversing the point of view, a simplicial complex $K$ is called triangulation of a topological space $X$, if $|K| = X$.
Note that for some purposes, as the definition of simplicial homology groups later on, only the combinatorial data of how the simplices are glued is relevant. A logical generalization of simplicial complexes, containing only their combinatorial data, are abstract simplicial complexes.

**1.6.3 Definition.** Let $V$ be a finite set. An abstract simplicial complex $K$ is a set of nonempty subsets of $V$, called simplices, such that

1. for every $v \in V$, it holds $\{v\} \in K$
2. if $s \in K$, and $s' \subseteq s$, then $s' \in K$.

In order to define the geometric realization of an abstract simplicial complex $K$, every element $s$ of $K$ is identified with a simplex $\tilde{s}$ in Euclidean space, such that every vertex of $K$ is in one-to-one correspondence with vertices of the simplices $\tilde{s}$. Then the geometric realization of $K$ is defined as the geometric realization of the simplicial complex consisting of the simplices $\tilde{s}$. This construction is unique up to homeomorphism. In the following, homology theory is developed for simplicial complexes, but everything works analogously for abstract simplicial complexes as well.
In order to analyze the structure of simplicial complexes, the $n$-simplices are given an orientation. Every linear order on $V$ induces an orientation of the simplices $[v_0, \ldots, v_n]$. Oriented simplices are denoted by $<v_0, \ldots, v_n>$. Two oriented simplices $<v_0, \ldots, v_n>$ and $<v_{\pi(0)}, \ldots, v_{\pi(n)}>$ are equivalent, iff $\pi$ is an even permutation. Hence, each $n$-simplex with $n \geq 2$ can carry exactly two orientations. The oriented simplices of the same dimension $n$ of a simplicial complex $K$ are collected in the set $K_n$.
Oriented simplicial complexes are arranged in abelian groups in the following way:

**1.6.4 Definition.** The free abelian group $C_n = C(K_n)$ generated of a finite set $K_n$ of oriented simplices is defined as

$$C_n = \left\{ \sum_{s \in K_n} k_s s \mid k_s \in \mathbb{Z} \text{ for all } s \in K_n \right\} \tag{1.63}$$

The sums in the above definition are understood in a formal sense independent of the structure of the simplicial complex $K$, adding $c = \sum_{s \in K_n} k_s s$ and $\tilde{c} = \sum_{s \in K_n} \tilde{k}_s s \in C_n$ is defined by $c + \tilde{c} = \sum_{s \in K_n} (k_s + \tilde{k}_s) s$.

These simplicial chain groups are related each other by defining group homomorphisms between them, mapping a simplex to a linear combination of its faces.

**1.6.5 Definition.** For an oriented simplicial complex $K$, the differential operator $\partial_n$ is defined as

$$\partial_n : C_n \to C_{n-1}, \qquad <v_0, \ldots, v_n> \; \mapsto \; \sum_{i=0}^{n} (-1)^i <v_0, \ldots, \hat{v}_i, \ldots, v_n> \tag{1.64}$$

and uniquely extended to a homomorphism. In this equation, $\hat{v}_i$ means deletion of vertex $v_i$.

It is common to denote chain groups and differentials by a sequence

$$\cdots \to C_{n+1} \overset{\partial_{n+1}}{\to} C_n \overset{\partial_n}{\to} C_{n-1} \to \ldots \tag{1.65}$$

and sometimes we omit the dimension index of the differential operator (we simply write $\partial_n = \partial$), since its construction is independent of the dimension of the chain group. An

easy computation shows, that $\mathrm{Im}(\partial_{n+1}) \subseteq \mathrm{Ker}(\partial_n)$, and therefore $\partial_n \circ \partial_{n+1} = 0$. On the other hand, kernels of differentials $\partial_n$ are in general not equal to images of $\partial_{n+1}$, this difference is measured by the homology groups of a simplicial complex.

**1.6.6 Definition.** The simplicial homology groups of an (abstract) simplicial complex are defined as

$$H_n(K) = \mathrm{Ker}(\partial_n)/\mathrm{Im}(\partial_{n+1}) \tag{1.66}$$

Homology groups fulfil several nice properties, which make them suitable to measure the structure of topological spaces. We want to recall

- *Independence of the triangulation:* If a topological space $X$ is triangulated by both $K$ and $\tilde{K}$, then $H_n(K) = H_n(\tilde{K})$. Note that this statement implicitly also says that homology groups are independent of the choice of orientation of the triangulating simplicial complex $K$.
- *Invariance property:* If two topological space $X$ and $\tilde{X}$ are homotopy equivalent [96, p.16], then $H_n(X) \cong H_n(\tilde{X})$. (The $\cong$ operator means isomorphic.)
- There are (at least) two other common notions of homology of a space $X$, namely cellular homology $H^C(X)$ and singular homology $H^S(X)$. As is proven for example in [56, p.128 and p.139], these homology theories are all isomorphic, $H_n(X) \cong H_n^C(X) \cong H_n^S(X)$. As a consequence, homology can be defined for all topological spaces without necessarily having a triangulation, and can be computed in various ways.

We want to interpret homology groups by giving two classical examples. The following example of the $k$-sphere motivates the interpretation, that the $n$-th homology group counts the number of n-dimensional holes.

**1.6.7 Example.** The homology groups of $\mathbb{S}^k$, the $k$-sphere, are given by

$$H_n(\mathbb{S}^k) = \begin{cases} \mathbb{Z} & k = 0, n \\ 0 & \text{otherwise.} \end{cases} \tag{1.67}$$

Closed orientable surfaces of genus $g$ possess $2g$ 1-dimensional holes:

**1.6.8 Example.** The homology groups of $T_g$, the closed orientable surface of genus $g$, are given by

$$H_n(T_g) = \begin{cases} \mathbb{Z} & n = 0, 2 \\ \mathbb{Z}^{2g} & n = 1 \\ 0 & \text{otherwise.} \end{cases} \tag{1.68}$$

In general, topological spaces with isomorphic homology groups need not be homotopy equivalent, and particularly not homeomorphic. But here, the following holds:

**1.6.9 Theorem** (Surface Classification Theorem)**.** *A closed surface with homology groups as in (1.68) is homeomorphic to $T_g$.*

*For a proof, see [82], Chapter 1.*

An important property of a simplicial complex $K$ is the Euler characteristic:

**1.6.10 Definition.** The Euler characteristic of a simplicial complex $K$ is defined as the alternating sum (compare [56, p. 146])

$$\chi(K) = \sum_{i=0}^{n} (-1)^i \, \#K_i \tag{1.69}$$

Here, # denotes the cardinality of a set.

The Euler characteristic can be computed over homology groups of the complex:

$$\chi(K) = \sum_{i=0}^{n} (-1)^i \text{ rank } H_i(K) \tag{1.70}$$

Since homology groups are independent of a special triangulation, the Euler characteristic is independent of a special triangulation as well. Given a triangulated, closed surface, the Euler characteristic can be used to determine its genus, since $\chi(K) = 2 - 2g$ holds.

### 1.6.2 Some tools to compute homology groups

In this subsection, we recall some important theorems on computation of homology groups.
Some theorems rely on the basic algebraic notion of an exact sequence.

**1.6.11 Definition.** An exact sequence is a sequence of groups $(A_n)_{n\in\mathbb{Z}}$ and homomorphisms $\partial_n : A_n \to A_{n-1}$, such that $\text{Ker}(\partial_n) = \text{Im}(\partial_{n+1})$. It is common to write

$$\cdots \to A_{n+1} \overset{\partial_{n+1}}{\to} A_n \overset{\partial_n}{\to} A_{n-1} \to \ldots, \tag{1.71}$$

and sometimes the mappings are even omitted.

Special attention is paid to short exact sequences.

**1.6.12 Definition.** A short exact sequence is an exact sequence of the form

$$0 \to A \to B \to C \to 0 \tag{1.72}$$

In the following, we want to list some important properties of short exact sequences:

- By definition, a short exact sequence $0 \to A \overset{s}{\to} B \overset{t}{\to} C \to 0$ splits, if there exits a homomorphism $r : C \to B$ such that $rt = \text{id}_C$. In such a case, we have a splitting

  $$B \cong A \oplus C, \tag{1.73}$$

  see [56, p.147]. If $C$ is a free abelian group, such a homomorphism $r$ always exists.

- The rank formula
$$\operatorname{rank}(B) = \operatorname{rank}(A) + \operatorname{rank}(C) \tag{1.74}$$
holds, [96, p.87].

Useful tools to compute the homology of a union of topological spaces are long exact sequences, the Mayer-Vietoris sequence and the excision theorem.
To start with, let $X$ be a topological space and $A \subseteq X$ a subspace. Relative homology groups $H_n(X, A)$ are defined by taking homology of the sequence

$$\cdots \to C_{n+1}(X)/C_{n+1}(A) \overset{\partial_{n+1}}{\to} C_n(X)/C_n(A) \overset{\partial_n}{\to} C_{n-1}(X)/C_{n-1}(A) \to \dots \tag{1.75}$$

The boundary operators are the quotient maps of the standard boundary operators, for simplicity denoted by the same symbol $\partial_n$. Then the following holds:

**1.6.13 Theorem** (Long Exact Sequence)**.** *Let $X$ be a topological space and $A \subseteq X$. Then the sequence*

$$\cdots \to H_n(A) \to H_n(X) \to H_n(X, A) \to H_{n-1}(A) \to \dots \tag{1.76}$$

*is exact [96, Thm 5.8].*

Note that for application of this theorem, the actual definition of the maps making the sequence exact (which we omit here) plays a minor role. Frequently, some groups in such a sequence equal zero and the long exact sequence breaks into short exact sequences.
Another frequently used sequence of this kind is the Mayer-Vietoris sequence. The Mayer-Vietoris sequence links the homology of subspaces $A, B \subset X$ to the homology of the union of their interiors $\mathring{A} \cup \mathring{B}$. It is stated in the following theorem.

**1.6.14 Theorem** (Mayer-Vietoris Sequence)**.** *Let $A, B \subseteq X$ such that $\mathring{A} \cup \mathring{B} = X$. Then the sequence*

$$\cdots \to H_n(A \cap B) \to H_n(A) \oplus H_n(B) \to H_n(X) \to H_{n-1}(A \cap B) \to \cdots \to H_0(X) \to 0 \tag{1.77}$$

*is exact [96, Thm 6.3].*

The main argument to prove the exactness of the Mayer-Vietoris sequence from the long exact sequence is the excision theorem, which is itself a useful tool in some homology computations.

**1.6.15 Theorem** (Excision)**.** *Let $X$ be a topological space and $Z \subseteq A \subseteq X$, such that $\overline{Z} \subseteq \mathring{A}$. Then the inclusion map of pairs $(X \backslash Z, A \backslash Z) \hookrightarrow (X, A)$ induces isomorphisms*

$$H_n(X \backslash Z, A \backslash Z) \to H_n(X, A) \quad \text{for all } n \in \mathbb{Z} \tag{1.78}$$

*For a proof, see [56, Thm 2.20].*

### 1.6.3 Proofs of the Main Theorems

In this subsection, we discuss two topological problems concerning spheres. In Chapters 3 and 4, results on these two problems are given and utilized without proof, which is preponed to this part of the thesis.

*Theorem 4.3.6*

Let $M = (V, E, F)$ be a 2-dimensional simplicial complex. Furthermore, let $M_b$ be the simplicial subcomplex of $M$ obtained by removing $k$ connected components from $M$ such that the geometric realization of $M_b$ is a surface with boundary consisting of $k$ simple closed polygons (see also Figure 4.1, top left). Let $C^1, \ldots, C^k$ be caps such that its boundaries consist of the boundary polygons of $M_b$ (see also Figure 4.3). Moreover, let $K$ be a handle for $C^1, \ldots, C^k$, so that $|\bigcup_{i=1}^{k} C^i \cup K| \approx \mathbb{S}^2$. Then

$$|M_b \cup K| \approx T_{k-1+g(M)}, \tag{1.79}$$

where $T_{k-1+g(M)}$ is the closed, orientable surface of genus $k - 1 + g(M)$ and $g(M)$ is the genus of $M$.

Proof.

It is obvious, that $|M_b \cup H|$ is a closed surface, so it remains to show that

$$g(M_b \cup H) = k - 1 + g(M).$$

The Mayer-Vietoris sequence (Theorem 1.6.14) for $M_b \cup H$ reads as

$$\cdots \to H_2(M_b) \oplus H_2(H) \to H_2(M_b \cup H) \to H_1(M_b \cap H) \to H_1(M_b) \oplus H_1(H) \to$$

$$H_1(M_b \cup H) \to H_0(M_b \cap H) \to H_0(M_b) \oplus H_0(H) \to H_0(M_b \cup H) \to 0.$$

In order to compute the terms in the sequence, note that the homology groups of a closed orientable surface $F$ of some genus $g(F)$ minus $k$ distinct points $p_1, \ldots, p_k$ are

$$H_* \left(F \backslash \{p_1, \ldots, p_k\}\right) = \begin{cases} \mathbb{Z} & * = 0 \\ \mathbb{Z}^{2g(F)+k-1} & * = 1 \\ 0 & * \geq 2 \end{cases}$$

This can be proven easily by excision,

$$H_* \left(F, F \backslash \{p_1, \ldots, p_k\}\right) \cong H_* \left(\mathbb{D}, \mathbb{D} \backslash \{p_1, \ldots, p_k\}\right)$$

and using the long exact sequence for $(F, F\backslash\{p_1, \ldots, p_k\})$.
Let us now have a closer look at the spaces occuring in the Mayer-Vietoris sequence for $M_b \cup H$:

- $M_b \cup H$ is homotopy equivalent to $\coprod_{i=1}^{k} \mathbb{S}^1$.
- $M_b$ is homotopy equivalent to a closed surface of some genus $g(M_b)$ minus $k$ points.
- $H$ is homotopy equivalent to a 2-sphere minus $k$ points by defintion of a handle.

Therefore, the Mayer-Vietoris sequence for $M_b \cup H$ computes to

$$0 \to \mathbb{Z} \to \mathbb{Z}^k \to \mathbb{Z}^{2g+k-1} \oplus \mathbb{Z}^{k-1} \to$$

$$H_1(M_b \cup H) \to \mathbb{Z}^k \to \mathbb{Z} \oplus \mathbb{Z} \to \mathbb{Z} \to 0.$$

Since $M_b \cup H$ is a closed surface, and because of the rank formula for abelian groups in exact sequences (Equation (1.74)), we deduce that

$$H_1(M_b \cup H) \cong \mathbb{Z}^{2(k-1+g(M))},$$

and therefore

$$g(M_b \cup H) = k - 1 + g(M).$$

□

*When is the geometric realization of a mesh homeomorphic to $\mathbb{S}^2$?*

The segmentation algorithms presented in Chapters 3 and 4 require a criterion which tells, when a given mesh is the geometric realization of a sphere. We give a little discussion here.

1. Given a mesh $M = (V, E, F)$.
   If the Euler charateristic (see 1.6.10) fulfils $\chi(M) = 2$ and
   for each $v \in V$, the link $\mathrm{Lk}(v)$ of $v$ is a simple, closed polygon then the geometric realization of $M$ is homeomorphic to a 2-sphere.

   Proof: The link condition ensures, that the mesh triangulates a closed surface. Since the Euler characteristic of this surface equals two, its genus is zero (because $\chi(M) = 2 - 2g(M)$) and therefore, $M$ triangulates a sphere.
   This criterion is especially suitable for computational purpose, since it can be checked in linear time, depending on the number of vertices of $M$.

2. Since the Euler characteristic of a closed surface can be computed from its homology groups (see1.70), Criterion 3.4.1 follows directly from the first item. Hence this is a second possible criterion to check, whether a mesh triangulates a sphere.

3. As was pointed out by H. Edelsbrunner (personal communication), Theorem 4.3.7 does not hold.

As a consequence, we recommend to use the first item instead of Theorem 4.3.7 in Chapter 4.

# 2 Segmenting surfaces of arbitrary topology: A two-step approach

Joint work with W. Hinterberger and O. Scherzer.

Published in Proceedings of SPIE, Ultrasonic Imaging and Signal Processing. Volume 6437, 2007.

This paper presents a topology adaptation system for parametric active contours in 3D. In a first step, the contour is discretized as a triangular mesh and evolved by evolution equations similar to (1.31). To incorporate topological adaptivity, intersecting triangles are detected and stopped in the evolution. In a second step, we show that from the evolved mesh and intersection data, a topologically transformed mesh can be computed which segments the object.

As main contribution of this paper, segmentation of 3D objects of unknown topology can be carried out in a stable and very efficient way.

External Contributions: W. Hinterberger provided parts of the numerical implementation (with which the results were computed) and O. Scherzer participated in discussions and revised the paper.

# SEGMENTING SURFACES OF ARBITRARY TOPOLOGY: A TWO-STEP APPROACH

Jochen Abhau [†], Walter Hinterberger [‡],
Otmar Scherzer [†,‡].

[†] Institut für Informatik,
Technikerstraße 21a, A-6020 Innsbruck
{jochen.abhau, otmar.scherzer}@uibk.ac.at.

[‡] IMCC Linz,
Altenbergstraße 69, A-4040 Linz; current address is
dTech-Steyr GmbH, Steyrerweg 2, A-4400 Steyr
Walter@mathconsult.co.at.

**Abstract.** We propose a two-step approach to segment closed surfaces in 3D of arbitrary topology. First, a pre segmentation step with an active contour method is performed. This evolution process does not take into account topology adaptions. Topologically correct segmentations are derived with Kazhdan's algorithm in a second step. Kazhdan's algorithm requires information on the surface normals, which are obtained from the active contour method. We show that the two-step algorithm is computationally efficient. Moreover, we apply the algorithms for segmentation of 3D ultrasound data.

**Key Words.** Image segmentation, active contour model, topology changes.

## 1. INTRODUCTION

Starting with the pioneering work [116], *active contour models* have been extensively studied and applied in many applications such as *image segmentation*, *surface reconstruction* and *shape modeling*.

In this paper we investigate parametric active contour (AC) models for segmentation of three dimensional data which allow for topological changes. There have been several contributions on topology adaptive active contours in 2D, some of them are reviewed below. However, there has been done much less work in 3D. In 2D *discrete active contours* are curves given by a discrete number of vertices which are connected by lines. The curves are called *snakes*. In [84], snakes are implemented topology-adaptive using an additional Freudenthal triangulation of the image plane. With this additional simplicial structure, a re parametrization (and therefore a topology adaption) is performed cyclically after a fixed finite number of iterations of the active contour evolution. The re parametrization is performed by computing the intersections of the (discretized) snake with the triangles of the Freudenthal triangulation and the snake is locally adapted to the topology. After

a re parametrization, the snake vertices are edges of the Freudenthal triangulation. A related approach is suggested in [16] where the discretized snake is a-priori restricted to have its snaxels on edges on an underlying two-dimensional grid. If the number of snaxels is large, already in 2D, the evolution is rather time-consuming. This is due to the fact that a vertex is not allowed to move further than the next gridpoint, and therefore many iterations are required.

AC models evolve a contour over time. The evolution is driven by the propagation speed $v_S$, which is specified a-priori and usually image content specific. We discretize the AC model in time and space by a sequence of triangular meshes (see e.g. [70]). Alternatively, one could discretize the surface by hexagons (see e.g. [37]) or by spline surfaces (see e.g. [77]).

We suggest and investigate the following segmentation procedure:

Initialization: The user provides a starting point inside the object to be segmented. Automatically, a sphere inside the object to be segmented with center at the starting point is generated.

Segmentation: During the evolution of AC model the sphere expands towards the boundary of the object until boundaries or self intersection (for instance topology changes) are detected. We perform segmentation, that is the boundary detection, borrowing the ideas of *intelligent scissors* (see [104]). Given the costs of moving between neighboring pixels (for instance the absolute value of the directional gradient between two neighboring pixels), the intelligent scissors calculate the minimal costs of moving from a user specified center to all other points. The segmentation is the closest contour to the center where all pixels exceed a cost threshold. Intelligent scissors are used to segment objects of different topology, for instance objects with inclusions. In our AC model we use a cost model, where the costs are evaluated along trajectories during the evalution. These costs are certainly higher than for intelligent scissors but it can be implemented more efficiently in the context of AC models. The advantage of intelligent scissors is that it can be implement more stable. Since the step length of the AC model is not restricted by the grid size of an underlying simplical structure, our algorithm is computational efficient.

Post Processing: The contour is re parametrized using Kazhdan's algorithm and topology adapted.

The paper is organized as follows: the active contour segmentation model with speed function motivated from intelligent scissors is derived in Section 2. Section 2 is concerned with the numerical implementation of the AC model. Kazhdan's algorithm for surface reconstruction is described in the Section 2. Moreover, we present some numerical experiments for segmentation of 3D clinical image data.

## 2. THE ACTIVE CONTOUR MODEL IN 3D

Let $I \in C_0^1(\Omega, \mathbb{R})$ be a representation of intensities of an image with image domain $\Omega \subset \mathbb{R}^3$.

The continuous active contour model (see [110]) consists in calculating a sequence of parametrized surfaces

$$(S_t : \Gamma \to \Omega)_{t \geq 0},\ \Gamma \subset \mathbb{R}^2$$

which for $t \to \infty$ approximates the segment of the object of interest. Following [29] we use physical considerations and derive an evolutionary partial differential equation where a surface evolves like a rubber skin of a balloon which is blown up. We use the *momentum equation*

$$\frac{\partial S_t}{\partial t} = f_t \text{ on } \Gamma\ . \tag{2.1}$$

The forcing term is modelled as $f_t = k_t(f_t^{int} + f_t^{ext})$:

1. $f^{int}$ describes an *internal force*, which physically simulates a steady air flow into the balloon. Neglecting other forces this results in a steady expansion in normal direction to the surface

   $$f_t^{int} = \vec{n}_t\ .$$

   Here $\vec{n}_t$ denotes the outer unit normal vector to the surface $S_t$.

2. An *external force*, the surface tension, which depends on the shape of the balloon and is given by

   $$f_t^{int} = \Delta S_t.$$

   This simulates the forces on the rubber skin. For segmentation application this enforces a smooth surface.

3. The indicator function

   $$k_t : \Gamma \to \{0, 1\}\,,$$

   is used to ensure that the contour does not move across edges and corners. For modelling $k_t$ we use a *boundary* indicator $\Psi_t^{cost}$ for the image, a *smoothness* indicator of the surface $\Psi_t^{curv}$ and an indicator for *self intersections*. Let $(u, v) \in \Gamma$, then in case $\Psi_t^{cost}(u, v)$ and $\Psi_t^{curv}(u, v)$ do not exceed a certain threshold we assume that there do not occur self intersections, and we set $k_t(u, v) = 1$, otherwise we put $k_t(u, v) = 0$. The later enforces termination of the evolution at $(u, v) \in \Gamma$.

   We use the following *boundary indicator* function

   $$\Psi_t^{cost} : \Gamma \to \mathbb{R},$$
   $$(u, v) \to \int_0^t \max\left\{0, \left\langle \nabla I(S_r), \frac{\partial S_r}{\partial t} \right\rangle (u, v)\right\} dr$$

   which is motivated from segmentation with *intelligent scissors* [104]. Intelligent scissors are hybrid indicator functions combining region based and gradient based

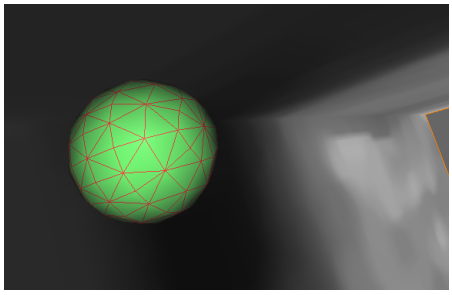


Figure 2.1: A small triangulated sphere is placed inside the object of interest.

segmentation. A typical examples of gradient based segmentation is Canny's edge detector [24] and an example of a region based segmentation technique is Mumford-Shah segmentation [87]. The functional $\Psi^{cost}$ takes into account the sum of absolute values (costs) of all gradients along a trajectory. This is different to intelligent scissors where the path with minimal costs is selected, while we follow the path of evolution of the surface. We restrict our attention to the evolution along trajectories of the surface evolution since this allows 3D real time segmentation. We observed, that for our application of filtering ultrasound data (see below) this approach is more stable than pure gradient based segmentation.

The smoothness control of the contour is achieved with

$$\Psi_t^{curv} = \mathcal{H}_t,$$

where $\mathcal{H}_t$ denotes the *mean curvature* of the surface $S_t$.

The contour moves outwards until internal and external forces balance, or $k_t$ becomes 0.

Discretizations of the forcing terms can be compared with established AC models in the literature, like for instance [29] and [84]; there the external forces are interpreted as spring forces (the forces serve as edge indicators and are related to the strength of an edge).

# 3. NUMERICAL IMPLEMENTATION OF THE BALLOON MODEL

Initially, we use a regularly triangulated sphere $S^0$ in $\mathbb{R}^3$ with 102 vertices and 200 triangles centered around a user supplied point, which is completely contained in the object of interest. Let $P$ be a vertex of the mesh of the contour $S_t$.

In the numerical implementation the derivatives in the active contour model (2.1) are approximated as follows:

- We use the following approximation of the time derivatives in the evolustion process:
$$\frac{\partial S_t}{\partial t}(u,v) = \frac{1}{h}(S^{n+1}(u,v) - S^n(u,v)))$$
if
$$S^n(u,v) = S_{hn}(u,v).$$
That is, we use an explicit Euler method. An explicit Euler method is considered to be inefficient in terms of numbers of iterations, however for this particular application where the cost functional is dependent on the trajectory, (semi-)implicit methods are inpractical.

- The Laplacian of the surface at $P = S^n(u,v)$ is approximated by the *umbrella* vector (see [66]):
$$U(P) = \frac{1}{|\mathcal{N}_V(P)|} \sum_{Q \in \mathcal{N}_V(P)} (Q - P),$$
where $\mathcal{N}_V(P) = \{\text{neighbor vertices of } P\}$.

- The normal vector to the triangulated surface at $P$ is approximated by the normalized mean of normal vectors of the neighboring triangles:
$$\vec{n}^n(P) = \left( \left| \sum_{\Delta \in \mathcal{N}_T(P)} \vec{n}^n_\Delta \right| \right)^{-1} \sum_{\Delta \in \mathcal{N}_T(P)} \vec{n}^n_\Delta, \tag{2.2}$$
where $\vec{n}^n_\Delta$ is the outer normal vector of the triangle $\Delta$ and $\mathcal{N}_T(P)$ is the set of triangles with vertex $P$. The normal vector is unique up to a sign. If in addition the mesh is given an orientation at $t = 0$, the orientation is preserved as long as no topology changes occur, and the outer normal vectors are uniquely specified during the evolution.

- The boundary indicator function $\Psi^{cost}$ is approximated as follows:
$$\Psi^{n,cost} = \frac{1}{n} \left( \sum_{i=1}^{n} \max \left\{ 0, \left\langle \nabla_h I(S^n), \frac{\partial S^n}{\partial t} \right\rangle \right\} \right)$$
where $\nabla_h I$ is a finite difference approximation of gradient of $I$. Let $P$ and $Q$ be two vertices of the mesh, then for each point $R$ on the edge
$$e_{P,Q} = \{tP + (1-t)Q | 0 < t < 1\},$$
the mean curvature $\mathcal{H}_t$ can be approximated by (see [105])
$$\mathcal{H}_t(R) = |(Q - P) \times \vec{n}^{(n)}_{\Delta_1} - (Q - P) \times \vec{n}^{(n)}_{\Delta_2}|$$
Here, $\vec{n}^{(n)}_{\Delta_i}$ denote the outer unit normals on the two adjacent triangles to $e_{P,Q}$.

During the numerical evaluation of AC we store an array $A_t$, which marks all voxels inside the contour at time $t$. Using this array we can detect self intersections by checking, if the next evolution step will move a mesh point into a marked voxel.

Using the above numerical approximations the evolution of the nodes is given by

$$S^{(n+1)} = S^{(n)} + hk^{(n)}(\lambda U(S^{(n)}) + \mu \vec{n}^{(n)}) \text{ on } \Gamma.$$

In our numerical implementation, we additionally implemented a re meshing scheme (refinement and coarsening) to get regular triangles during the iteration. This ensures higher accuracy of the umbrella vector approximating the surface Laplacian and of the normal at the vertex points. Both approximations are inaccurate in case of obtuse triangles or highly varying triangle sizes at a vertex.

Refinement and coarsening is implemented as follows: If the length of an edge $e$ exceeds a certain user-defined parameter, a refinement of the two neighbor triangles is performed. Vice versa, it if the edge length is smaller that another threshold then coarsening is perormed. The two algorithms read as follows:

Algorithm: Refinement

Given triangular mesh $(V, E, F)$ and lower length bound $\epsilon$.

WHILE $\{e \in E \mid \text{length}(e) > \epsilon\} \neq \emptyset$
  $S, T \leftarrow$ endpoints$(e)$
  $P, Q \leftarrow$ opposite vertices$(e)$
  $R \leftarrow \frac{1}{2}(S + T)$
  $V \leftarrow V \cup \{R\}$
  $E \leftarrow E \backslash \{(S, T)\}$
  $E \leftarrow E \cup \{(P, R), (R, Q), (S, R), (R, T)\}$
  $F \leftarrow F \backslash \{(P, S, T), (Q, T, S)\}$
  $F \leftarrow F \cup \{(P, R, S), (Q, S, R), (T, R, P), (Q, R, T)\}$
END WHILE

Algorithm Coarsening

Given a mesh $(V, E, F)$ and upper distance bound $\epsilon$.

WHILE $\{e \in E \mid \text{length}(e) < \epsilon\} \neq \emptyset$
  $P, Q \leftarrow$ endpoints$(e)$
  $S, T \leftarrow$ opposite vertices$(e)$
  $R \leftarrow \frac{1}{2}(P + Q)$
  $V \leftarrow V \backslash \{P, Q\}$
  $V \leftarrow V \cup \{R\}$
  $E \leftarrow E \backslash \{(S, P), (S, Q), (P, T), (Q, T), (P, Q)\}$
  $E \leftarrow E \cup \{(S, R), (R, T)\}$
  $F \leftarrow F \backslash \{(S, P, Q), (T, Q, P)\}$
  replace coordinates $P$ and $Q$ by $R$ in each triangle
END WHILE

Some results of the active contour model are presented in Figure 2.4.

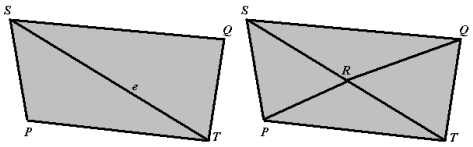



Figure 2.2: If the edge $e$ appears to be too long, vertex $R$ and lines $PR$, $RQ$ are inserted to subdivide the triangles.

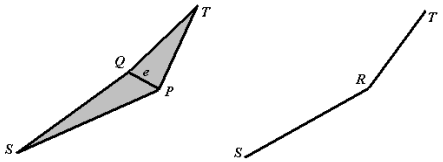



Figure 2.3: If two points are closed, they are melted.

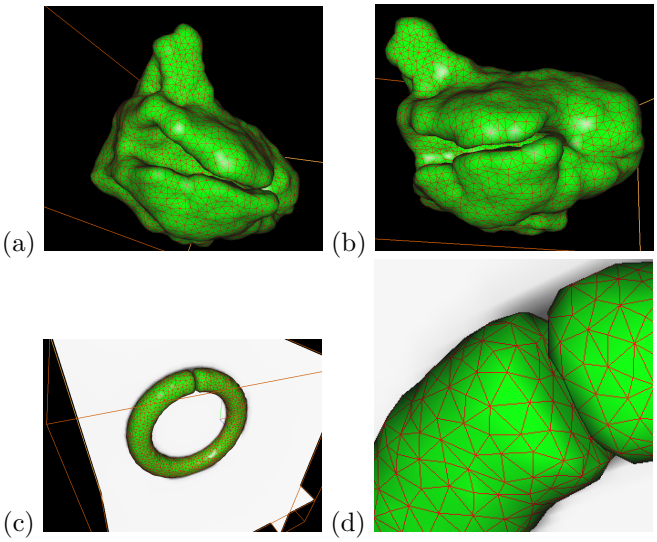



Figure 2.4: In (a) and (b), the end of the balloon evolution is shown for the cyst example. The evolution result for the torus can be seen in (c) and (d). All these four meshes are still homeomorphic to a sphere.

# 4. THIRD STEP: TOPOLOGY CHANGES

The AC contour model described in the previous section did not allow for topology changes. In fact whenever topology changes are predicted, the iteration is locally terminated. Therefore the result of the AC model the balloon evolution is homeomorphic to a sphere. An example for the result of the AC model is shown in figure 2.4.

In the following we present an algorithm that allows to reconstruct the correct topology of surfaces of genus $g > 0$ from the output of the AC model.

Let $P_1, \ldots, P_{N_{term}}$ denote the mesh points of the stationary surface of the AC model. Moreover, we denote by $\vec{n}_1, \ldots, \vec{n}_{N_{term}}$ the weighted normals at the mesh points, computed with formula (2.2). In practice the informations on the mesh points and their weighted normals is sufficient to uniquely determine the mesh surface. However, there can occur situations when the mesh is not uniquely determined, which in a mathematical notation means that for each mesh point $P_i$ there exists a scalar $\mu_i$ such that

$$\sum_{\Delta \in \mathcal{N}_T^1(P_i)} \vec{n}_\Delta = \mu_i \sum_{\Delta \in \mathcal{N}_T^2(P_i)} \vec{n}_\Delta. \tag{2.3}$$

The set of configurations $\{P_1, \ldots, P_{N_{term}}\}$ admitting such a relation is a finite intersection of a finite union of codimension-2-subspaces of $\mathbb{R}^{3N_{term}}$ and hence has Lebesgue measure zero. If the mesh after the balloon evolution encloses a flat region, the upper mentioned ambiguities do not influence the shape of the mesh. Otherwise we can think of the vertices as a random sample of surface points, such that the upper relation is not satisfied. These considerations motivate to use as mesh information just points and normals.

Given the mesh after the balloon evolution, to approximate the surface in a topological correct form, we distinguish beetween two types of vertices:

1. The flow of a vertex has been stopped during the AC evolution since a self intersection has been detected. Such a vertex must be discarded as an interior point of the object.

2. In all the other cases, a vertex is considered to be part of the boundary of the object.

The following procedure shows how to remove vertices of type 1 which belong to the interior of the object. For this purpose the mesh is first opened and afterwards remeshed.

## Opening of mesh for topology adaption

A procedure for opening and updating of the normal is as follows:

- [1] Mark all triangles which contain a vertex of type 1.

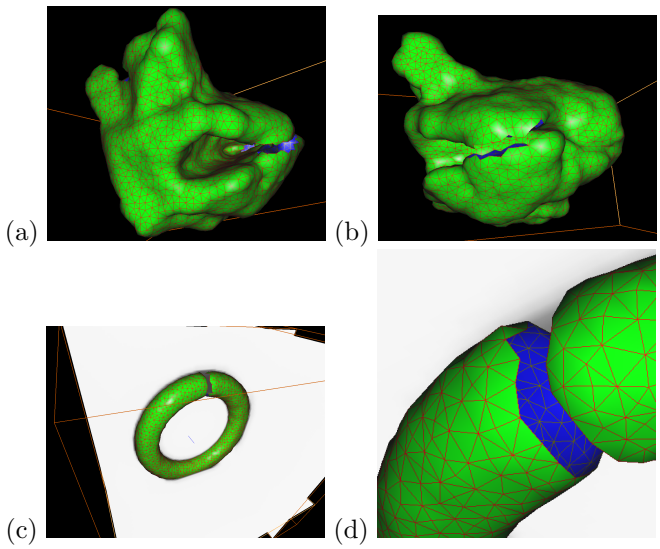



Figure 2.5: Opening Algorithm: vertices and triangles at places of self-intersection are discarded. The remaining vertices and triangles are remeshed.

- [2] For all vertices $P$ contained in a marked triangle, update the surface normals: Set $\mathcal{N}_T(P_i)) = \{\text{triangles incident to } P_i, \text{ not marked}\}$, and

$$n_{P_i} = \left( \left| \sum_{\Delta \in \mathcal{N}_T(P_i)} \vec{n}_\Delta \right| \right)^{-1} \sum_{\Delta \in \mathcal{N}_T(P_i)} \vec{n}_\Delta$$

- [3] Delete all marked triangles and vertices of type 1 from the mesh.

We obtain a meshed surface with boundaries and update the normals of the vertices which had a neighboring triangle which has been cut out.

The result of the opening procedure is shown in figure 2.5.

## The remeshing algorithm

We discard all the edges of the mesh after the opening procedure, such that only surface points and their normals remain.

We compute the topology adapted mesh with vertices and outer unit normals following the mesh reconstruction method described in [61]. To be self-contained, we shortly summarize this algorithm, the features and adapt it to our purposes.

Let $P_1, \ldots, P_N$ denote the remaining points on the surface after the opening procedure, and $\vec{n}_1, \ldots, \vec{n}_N$ the corresponding outer unit normals. Moreover, let $M \subset \mathbb{R}^3$ denote the object of interest. We approximate the indicator (or characteristic) function $1_M$ of the

set $M$ by computing the fourier series expansion of $1_M$. For this purpose, let

$$\hat{1}_M(k) = \int_M e^{-i\langle k,x\rangle} dx$$

be the k-th Fourier coefficient of $1_M$, $k \in \mathbb{Z}^3$.
Moreover, for $x \in \mathbb{R}^3$, let

$$F_k(x) = \begin{pmatrix} \frac{ik_1}{|k|^2} e^{-i\langle k,x\rangle} \\ \frac{ik_2}{|k|^2} e^{-i\langle k,x\rangle} \\ \frac{ik_3}{|k|^2} e^{-i\langle k,x\rangle} \end{pmatrix}.$$

This function satisfies

$$\mathrm{div} F_k(x) = e^{-i\langle k,x\rangle}, k \in \mathbb{Z}.$$

Therefore, by Stokes' Theorem, we have

$$\int_M e^{-i\langle k,x\rangle} dx = \int_{\partial M} \langle F_k(p), \vec{n}(p)\rangle \, dp.$$

Therefore, by using Monte-Carlo-Approximation we find

$$\int_{\partial M} \langle F_k(p), n(p)\rangle \, dp \approx \frac{c(M)}{N} \sum_{i=1}^{N} \langle F_k(P_i), \vec{n}_i\rangle \, .$$

Here $c(M)$ denotes the surface area of $M$, a constant which actually need not be computed for what follows.

To summarize, we have shown that for every $x \in \mathbb{R}^3$

$$\begin{aligned} 1_M(x) &= \sum_{k\in\mathbb{Z}^3} \hat{1}_M(k) e^{i\langle k,x\rangle} \\ &\approx \text{const} \underbrace{\sum_{k\in\mathbb{Z}^3} \sum_{i=1}^{N} \langle F_k(P_i), \vec{n}_i\rangle}_{=:\tilde{1}_M(x)} \, . \end{aligned}$$

Following [61], an approximation of the characteristic function $1_M$ can be calculated using the mean value $\mu = E(\tilde{1}_M)$ and by appropriate thresholding

$$1_M(x) \approx \begin{cases} 1, & \tilde{1}_M(x) \geq \mu \\ 0, & \tilde{1}_M(x) < \mu. \end{cases}$$

Once the indicator function of the object has been computed, the surface mesh can be computed with the marching cubes algorithm, see [79].

Figure 2.6 illustrates the algorithm for topology adaption and the result obtained after applying the marching cube algorithm.

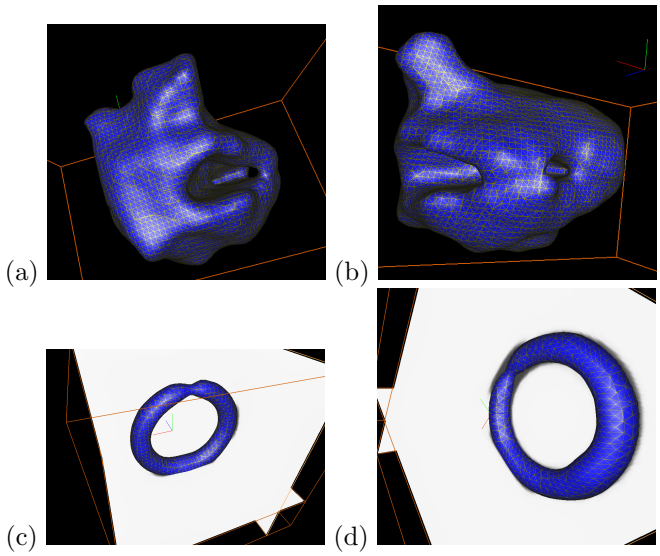


Figure 2.6: After remeshing, the mesh shown in (a) and (b) is obtained, (c) and (d) show the result for the torus. Note that both surfaces have genus 1, hence a topological change has taken place.

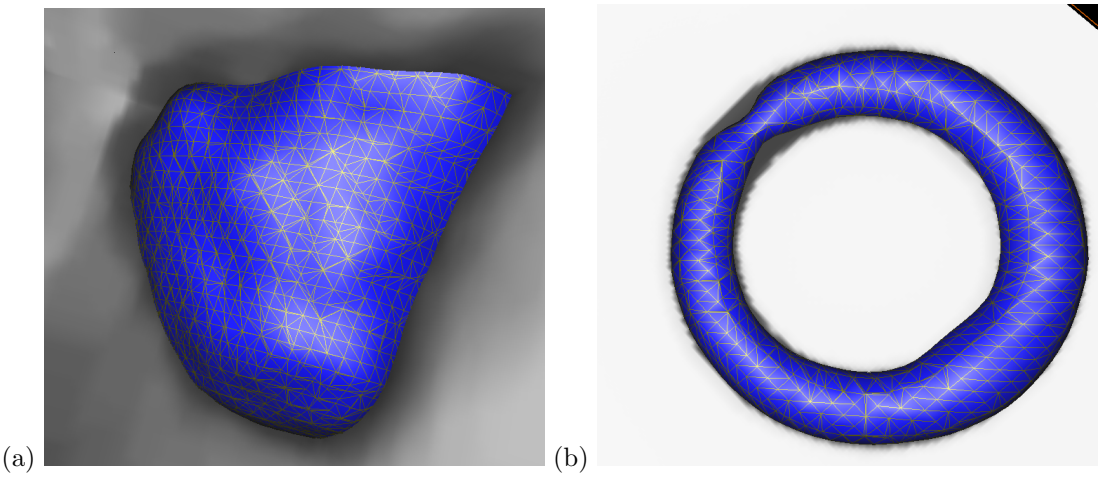


Figure 2.7: (a) Zoom into the cyst mesh. (b) The segmentation of the torus is equally good in regions where no topological change occured.

## 5. RESULTS AND DISCUSSION

We tested our algorithm with artificial and real 3D voxel data. Wo have considered two examples showing a cyst in a kidney and an artificial torus shown in Figures 2.1 and 2.4-2.7. During biopsy a needle has been sticked into the cyst. Since the needle penetrates the object, mathematically we might say that this object has genus 1. The sequence of pictures shows

- the original voxel image,
- the voxel image filtered (which is applied before segmentation to stabilize the algorithm),
- the small sphere is placed inside the object for initialization (is the single user interaction),
- the final result of the AC evolution (the surface is still homeomorphic to a sphere),
- the sphere points which were stopped because of threatening self-intersections are discarded,
- the surface mesh after reconstruction by normals and the marching cubes algorithm,
- the final segmentation result.

We computed the two examples on a Pentium 4 Computer with 2 GB RAM and 3,5 GHz CPU.

While Step 2 took approximately 8 seconds for the cyst and 5 seconds for the torus, the remeshing step 3 was done in ca. 0.1 seconds in both cases.

The drawback of our method can be seen in figure 2.7(b). In regions of topological change, the segmentation is a bit imprecise. This is a consequence of the curvature control. At the end of the evolution the deformed sphere fills the object up to small pieces at the boundary. Discarding intersection triangles, the normals of the resting triangles on the boundary of the opened mesh point slightly backwards, away from the boundary. Therefore, the torus is a bit thinner in this region. One possible solution to this problem might be a second usage of the first step, now blowing up the present (topologically correct) surface.

## 6. SUMMARY AND FUTURE WORK

In this paper we have proposed a new approach for active contours which allows for topology adaptations. The algorithms consists of two steps, and AC model which supresses topology changes and a topology adaptions using Kazhdans algorithm and a marching cube algorithm. The AC model is time efficient since it evolves a surface withour using

an underlying grid structure. In practical applications we applies the segmentation algorithm to filtered data. The topic of adequate filtering has not been addressed in this paper but is important to achieve relyable results. As a future work we intend to study adaptions of the contour where the topological adaptation has taken place.

## Acknowledgments

The work of J.A. and O.S. is supported by the Austrian Science Foundation (FWF) Projects Y-123INF; O.S. is additionally supported by projects FSP 9203-N12 and FSP 9207-N12. Part of the work of W.H. has been supported by the Austrian Ministry for Economy and Labour and by the Government of Upper Austria within the framework "Industrial Competence Centers".

We thank GE - Medical Systems, Kretztechnik, for providing the voxel image of the cyst and Tobias Riser for his QT-Viewer, with which the example pictures of the cyst and the torus were visualized. The QT viewer has been developed within the TWF project Parallelisierte Datenauswertung am HPC.

# 3 A robust and efficient method for topology adaptations in deformable models

Published in Proceedings of VISAPP, p.375-382, 2008.

# A ROBUST AND EFFICIENT METHOD FOR TOPOLOGY ADAPTATIONS IN DEFORMABLE MODELS

Jochen Abhau [†].

[†] Institut für Informatik, Leopold-Franzens-Universität,
Technikerstraße 21a, Innsbruck, Austria
jochen.abhau@uibk.ac.at.

**Abstract.** In this paper, we present a novel algorithm for calculating topological adaptations in explicit evolutions of surface meshes in 3D. Our topological adaptation system consists of two main ingredients: A spatial hashing technique is used to detect mesh self-collisions during the evolution. Its expected running time is linear with respect to the number of vertices. A database consisting of possible topology changes is developed in the mathematical framework of homology theory. This database allows for fast and robust topology adaptation during a mesh evolution. The algorithm works without mesh reparametrizations, global mesh smoothness assumptions or vertex sampling density conditions, making it suitable for robust, near real-time application. Furthermore, it can be integrated into existing mesh evolutions easily. Numerical examples from medical imaging are given.

**Key Words.** Segmentation, deformable model, topology adaptation, homology theory, medical image analysis.

## 1. INTRODUCTION

Since the pioneering work [116], deformable models have been used very successfully in the areas of computer vision and pattern recognition. In general, one can differ between two classes of deformable models: Explicit or parametric models, and implicit ones.

Implicit models, i.e. level-set techniques, were introduced in [89] and further developments were done in [25]. Since the contour is given as the isosurface of a scalar function, topology adaptations are handled naturally in implicit models. Nevertheless, explicit models are often preferred. This is due to the fact, that the mathematical equations are sometimes easier to formulate, and user interaction and special geometrical constraints can be incorporated easily. However, topological transformations are difficult to implement in explicit (2D or 3D) contour evolutions.

In [84], evolving polygons in 2D are made topology-adaptive by using a Freudenthal triangulation of the image plane. A reparametrization is performed cyclically after a fixed finite number of iterations of the polygonal evolution by intersecting the poly-

gon with the Freudenthal triangles. Polygon self-intersections are detected inside each Freudenthal triangle, and topology adaptations - if necessary - are obtained by some case distinctions. Similar ideas have already been developed in [37]. In [17], the mesh is a-priori restricted to have its vertices on edges on an underlying two-dimensional grid. If the number of vertices is large, already in 2D, the evolution is rather time-consuming. This is due to the fact that vertices must move grid point by grid point. In [70], special restrictions on edge lengths and angles in the mesh are imposed to detect self-collisions and to adapt topology in 3D mesh evolutions. Some progress has been made in [72], [73] and [71] allowing for less mesh vertices. Nevertheless, global mesh restrictions remain which have to be controlled in every iteration step, slowing down the evolution. Furthermore, self-collisions of the mesh are checked by a distance field which is time-consuming and not fast enough for near real time applications, see also [111]. A two-step topology adaptive algorithm has been proposed in [3], where a standard active contour evolution is performed first, and topology is adapted by a postprocessing step afterwards. In [93], after each evolution step the mesh is retriangulated by a restricted Delaunay triangulation in $O(n \log n)$ time, $n$ being the number of mesh vertices. As is reported there, the performance of this algorithm is in the range of [71].

To summarize, most of the algorithms mentioned above work well in 2D, but in 3D they tend to be very time-inefficient, because of the additional structure to be updated and maintained. Furthermore, the topology adaptation systems have effects on the mesh evolution and do not run independently.

We propose a topology adaptive active contour algorithm which bases on two novel ingredients:

- For collision detection during mesh evolution, the image space is subdivided into small axis-aligned bounding boxes. Every mesh vertex is mapped to a hash index, depending on the bounding box it lies in. Intersection tests are performed between triangles which contain vertices of the same hash index. By choosing the hash function and the bounding boxes appropriately, this algorithm runs linear in the number of vertices.

- For topology adaptation, we delete colliding mesh parts. The boundaries of these mesh parts are reconnected by using a database which consists of reasonable connections. This database is mainly derived from homology theory. During mesh evolution, possible mesh reconnections are looked up from the database and by a few triangle-triangle intersection tests, we decide which one to take. Since topology adaptations are always performed locally, the running time for collision detection dominates, and therefore the running time for the whole topology adaptation system is linear in the number of mesh vertices.

## Outline

This paper is organized as follows: In Section 3, we give a general decription of our novel topology adaptation algorithm. In Section 3 we present the collision detection system.

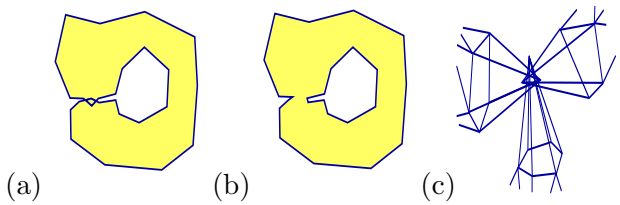



Figure 3.1: In (a), a surface (yellow) has evolved around the nested ball in the middle. A small tube connects the nested ball and the outer object. Edges of this tube self-intersect. The tube is cut in (b). For sake of clarity, only the 2D projections are drawn. In (c), three mesh parts collide. A handle has to be inserted between these parts to adapt topology.

In the next section, we explain the generation of a database used for topology adaptations. In Section 3, we summarize the topology adaptive mesh evolution algorithm. In Section 3, we provide and discuss experimental results of the new method in 3D image segmentation. Section 3 concludes the paper.

## 2. GENERAL DESCRIPTION OF OUR ALGORITHM

Our algorithm is designed for multiple connected surfaces in 3D and consists of the following steps: An active contour model is used to evolve a mesh until mesh self-collisions are detected. The topology adaptation is performed and afterwards the active contour evolution is further continued. In this paper we focus on algorithms for detection of self collisions and topology adaptations. Active contour models are not discussed further and can for instance be found in [29].

*Collision detection* is performed by a spatial hashing algorithm, motivated by [111]: A hash function is used to index each mesh vertex according to its position relative to small axis aligned bounding boxes. Triangles having vertices with the same hash index are checked for intersection. Since this test can be performed by iterating through hash indices, the expected running time for the collision detection algorithm is linear in the number of vertices and the hash table size.

*Topology adaptation* is mainly performed by a precalculated database of topology changes derived from homology theory:
The collision detection algorithm computes intersecting triangles. Triangles can collide during an evolution because of two reasons:

(1) A nested object has been detected, i.e. there is a smaller object enclosed by a larger one, see Figure 3.1(a).

(2) Two or more mesh parts are actually forming a handle, see Figure 3.1(c).

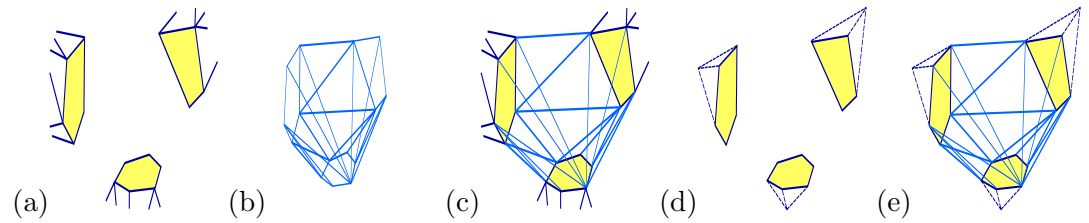



Figure 3.2: In (a), a mesh with three open holes (in yellow) surrounded by simple, closed polygons is shown. It results from deleting the overlapping mesh parts in Figure 3.1(c). A possible handle for mesh retriangulation is shown in (b), inserting the handle results in (c). The same handle can be used to triangulate the caps shown in (d) to a topological sphere in (e).

These two cases can be distinguished by analyzing the positions of the vertices which belong to the intersecting triangles. Case (1) is treated by cutting the small tube shown in Figure 3.1(b). In case (2) we delete the overlapping triangles, and after some preprocessing, we obtain a mesh with $k$ holes surrounded by simple, closed polygons as demonstrated in Figure 3.2(a). Among all sets of edges and faces $(E, F)$ connecting the mesh parts (call them *handles*), we are looking for a handle $(E_0, F_0)$ such that

(a) $(E_0, F_0)$ produces a *combinatorially consistent* mesh, i.e.
  (i) every edge belongs to the boundary of exactly two triangles
  (ii) every triangle borders exactly three triangles
  (iii) all mesh parts are connected to each other

(b) the edges and faces of $(E_0, F_0)$ produce an *intersection-free* adapted mesh.

It turns out, that condition (a) is just dependent on $k$ and the number of vertices in each of the $k$ closed polygons, and independent of the vertex coordinates. We use this observation for computation of a database of handles *before any mesh evolution*. There, for every realistic $k$ and realistic vertex numbers of the $k$ polygons, handles fulfilling condition (a) are stored. Now *during* a mesh evolution, a topology change is simply performed by looking up handles fulfilling (a) in the database, and choosing one which fulfils (b). This can be done very efficiently by a fast triangle-triangle intersection test.
*Beforehand generation of the handle database:*
Assume that we want to retriangulate $k$ open mesh parts consisting of $\lambda_1, \ldots, \lambda_k$ vertices, but that the vertex coordinates are unknown. Since the combinatorial consistency criterion (a) already makes sense in this (purely combinatorial) situation, we use it for computation of the database. In order to keep the actual number of stored handles small, we also study the structure of the database. Both for computation and for structure analysis, it is useful to reformulate the combinatorial consistency criterion in mathematical terms of homology groups. This is done by the following key observation, which is depicted in Figure 3.2: Handles for mesh retriangulation, as shown in Figure 3.2(b)-(c),

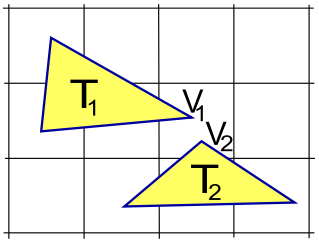



Figure 3.3: Triangles $T_1$ and $T_2$ are checked for intersection, since vertices $V_1$ and $V_2$ are mapped to the same hash key.

are the same as handles which connect caps to spheres, as shown in Figure 3.2(d)-(e). The caps are obtained from the open mesh parts. This observation allows for database generation by standard homology software, and the database structure can be described easily by actions of the symmetric group which leave the homology invariant.

# 3. THE (SELF-)COLLISION DETECTION SYSTEM

In order to detect self-collisions of one mesh component as well as collisions between two or more components, we use a spatial hashing approach.

We are given a triangular mesh $M = (V, E, F)$ in a bounded region $\Omega \subset \mathbb{R}^3$. As in [111], the collision detection algorithm subdivides $\Omega$ into small axis aligned bounding boxes. In a single pass, all vertices $v = (v_x, v_y, v_z) \in \Omega$ are mapped to hash indices by a function

$$\begin{aligned} &\text{hash} : V \to \{0, \ldots, m-1\}, \\ &v \mapsto \lfloor v_x/l \rfloor \, p_1 + \lfloor v_x/l \rfloor \, p_2 + \lfloor v_x/l \rfloor \, p_3 \bmod m \end{aligned} \tag{3.1}$$

Here, $l$ is a parameter for the box size. The coordinates are scaled by $l$ and rounded down to the next integer. In image segmentation, we usually set the box size parameter to 1, such that a box is given by one voxel. The $p_i$ are large prime numbers, and $m$ indicates the hash table size. In a second pass, for each hash index $i \in \{0, \ldots, m-1\}$, the algorithm processes the vertices with hash index $i$. First, the vertices are gathered to connected components, where two vertices are connected, if they are neighbors, i.e. there is a mesh edge connecting them. Each two triangles adjacent to vertices of different components are intersected by a fast triangle-triangle intersection test, f.e. [86]. If an intersection between two triangles is detected, we store the adjacent component vertices as (self-)intersection data. For an illustration, see Figure 3.3. We have optimized the parameters of the spatial hashing algorithm in order to obtain maximum speed. As [111] reports, spatial hashing with tetrahedral meshes works best if the hash table size is chosen approximately equal to the number of mesh vertices. In numerical experiments with our triangle-triangle intersection test, we found out that the number of hash indices chosen to be twice the number of vertices is appropriate for triangle meshes. With this

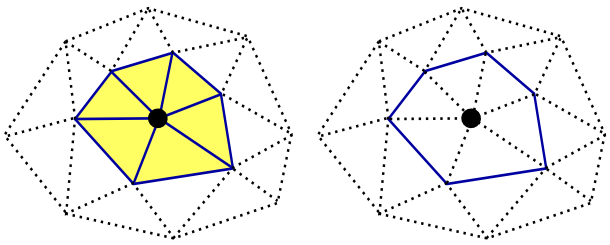


Figure 3.4: Star and link of a vertex. Here, the link is a simple, closed polygon.

choice, only a very few number of hash collisions occurs, and the complexity of processing the vertices of one hash index can be regarded as constant. Therefore, the expected time for detecting (self-)collisions of the mesh is linear, the memory usage is small (twice the number of vertices) and the required data structures are simple. Note that in most applications with a sufficiently smooth mesh, only a very small number of colliding vertices occurs.

# 4. GENERATION OF THE HANDLE DATABASE

Assume we are given $k$ simple closed polygons consisting of $\lambda_1, \ldots, \lambda_k$ vertices. The considerations in this section are independent of the vertex coordinates. As explained in Section 3 and Figure 3.2, the database consists of handles $(E, F)$ for the data $(k; \lambda_1, \ldots, \lambda_k)$, which produce a combinatorially consistent mesh. These handles can be computed by forming caps out of the simple, closed polygons and looking for triangulations of the caps to a single sphere (as in Figure 3.2(d)-(e)). Therefore, we first formulate a topological characterization of 2-spheres in Subsection 3, which allows for easy computation of the handle database and examination of its structure in Subsection 3.

## 4.1. Topological Characterization of 2-Spheres

We introduce some required topological notions here, details can be found f.e. in [39] and [56], Chapter 2. We are given a mesh $M = (V, E, F)$ embedded in $\mathbb{R}^3$.
A measure of connectivity of $M$ are its *homology groups* $H_0(M)$, $H_1(M)$ and $H_2(M)$, indicating the number of connected components, tunnels and voids of $M$. As an important example, if $M$ is a triangulation of the 2-sphere $\mathbb{S}^2$,

$$H_i(M) = \begin{cases} \mathbb{Z} & i = 0, 2 \\ 0 & i = 1. \end{cases} \tag{3.2}$$

For a vertex $v \in V$, *Star* and *Link* are defined by

$$\mathrm{St}(v) = \{\tau \in M \mid \text{exists } \tau' \in M \text{ with } v \subseteq \tau',\ \tau \subseteq \tau'\}, \tag{3.3}$$

$$\mathrm{Lk}(v) := \{\tau \in \mathrm{St}(v) \mid \tau \cap v = \emptyset\}. \tag{3.4}$$

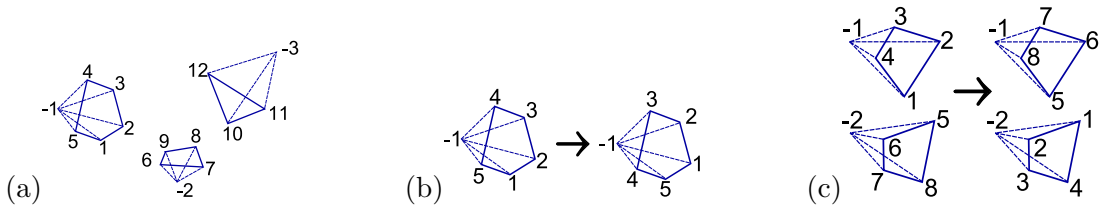



Figure 3.5: In (a), cap vertices are numbered. Group actions are shown in the next two diagrams: A rotation is shown in (b), an exchange is shown in (c).

Star and link are visualized in Figure 3.4.
With these notions, we can characterize a 2-sphere:

**3.4.1 Criterion.** Given a mesh $M = (V, E, F)$. If

(1)
$$H_i(M) = \begin{cases} \mathbb{Z} & i = 0, 2 \\ 0 & i = 1 \end{cases} \tag{3.5}$$

and

(2) for each $v \in V$, $\mathrm{Lk}(v)$ is a simple, closed polygon

then $M$ triangulates a 2-sphere.

This criterion follows easily from the fact, that a mesh triangulates a compact surface, if condition (2) is fulfilled, and that a compact surface with homology groups as in condition (1) is homeomorphic to a 2-sphere, [82], Chapter 1.

## 4.2. Computation and Structure of the Handle Database

We now use the sphere classification criterion 3.4.1 for construction of the handle database. Let us first compute the number of faces $f_h$ of a handle $(E, F)$. We set $\mu_l = \sum_{i=1}^{l} \lambda_i$ and number the vertices of the $k$ polygons by $1, \ldots, \mu_1; \ldots; \mu_{k-1}+1, \ldots \mu_k$. The (artificial) cap vertices are numbered by $-1, \ldots, -k$, see Figure 3.5(a). Let $v, e, f$ denote the number of vertices, edges and faces of the final sphere. From Euler's formula we deduce $v - e + f = 2$. In a triangulated sphere, each edge is the boundary of two triangles, and each face has three edges, which gives $2e = 3f$. Both formulas together give $f = 2v-4$, and from v=$\mu_k+k$ it follows $f = 2\mu_k+2k-4$. Since the caps contain $\mu_k$ faces altogether, and the sphere is constructed out of the caps, we obtain $f_h = \mu_k + 2k - 4$. For computation of handles, $f_h$ faces are generated and Criterion 3.4.1 is checked. We use [90] for the homology part of the criterion. As an example, in case $\lambda_1 = 5$, $\lambda_2 = 4$, $\lambda_3 = 3$, we obtain faces

(1,2,11) (1,5,12) (1,11,12) (2,3,8) (2,8,10) (2,10,11) (3,4,7)
(3,7,8) (4,5,6) (4,6,7) (5,6,9) (5,9,12) (8,9,10) (9,10,12)

and adjacent edges as handle. Altogether, we computed 120 handles in this case. It is not necessary to compute all handles by Criterion 3.4.1, since we can also compute handles from existing ones by application of *symmetric group actions* on the handles.

(1) *Rotations*: As the vertices of a polygon are cyclically ordered, the vertex which gets the start number $\mu_{i+1}$ is arbitrarily chosen. Therefore, these numbers can be rotated arbitrarily, as shown in Figure 3.5(b).

(2) *Exchanges*: Reflecting the fact, that on this combinatorial level two polygons with the same number of vertices $\lambda_i = \lambda_j$ cannot be distinguished, we note that two such neighborhoods can be exchanged. This is depicted in Figure 3.5(c).

As an example, rotating the handle of the previous example by the operation shown in Figure 3.5(b), gives another handle with faces

(5,1,11) (5,4,12) (5,11,12) (1,2,8) (1,8,10) (1,10,11) (2,3,7)
(2,7,8) (3,4,6) (3,6,7) (4,6,9) (4,9,12) (8,9,10) (9,10,12).

With this knowledge, the size of the database becomes very small, since only a few generating elements need to be stored, the others are obtained by applying group actions.

# 5. THE TOPOLOGY ADAPTATION SYSTEM

The main ingredients of the topological adaptation system are self-collision detection and the handle database, as described in Sections 3 and 3. Here we give the missing routines necessary for a complete, executable algorithm.

- *Make components of colliding vertices:* Assume that the self-collision detection algorithm has detected overlapping mesh regions, represented by non-neighboring vertices lying in the same axis aligned bounding box, which are adjacent to intersecting triangles. These vertices $v_i$ are grouped to connected components $C_1, \ldots, C_k$, such that for $i \neq j$, two arbitrary vertices $v \in C_i$, $w \in C_j$ have no common neighbor. This is done by initializing each set $C_i$ with the single element $v_i$, and as long as two sets $C_i$, $C_j$ have common neighbors, they are merged and these common neighbors are inserted to the union additionally.

- *Local refinement:* As a next step, we plan to remove all vertices of the sets $C_i$ (and adjacent pieces) from the mesh. Since the holes we obtain after removal are not always surrounded by simple, closed curves as required for the following steps, we perform a local refinement around the sets $C_i$ first. This procedure is shown in Figure 3.6. All edges between cluster vertices $C_i$ and vertices of $V \backslash C_i$ are subdivided by an additional vertex, and edges between the new vertices are inserted for triangulation, see Figure 3.6(b). After this refinement procedure, the vertices of the mesh $C_i$ are removed from the mesh.

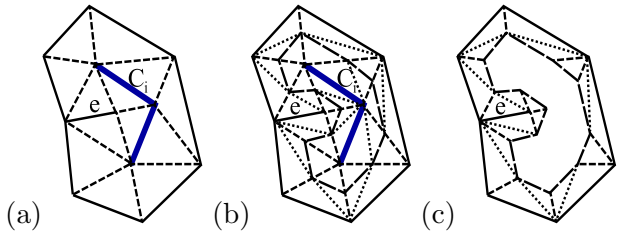



Figure 3.6: In (a), edge $e$ ver that removing $C_i$ leads to a simple closed polygon around the hole. A local refinement is performed in (b). After removal of $C_i$, a simple closed polygon around the hole arises in (c).

- *Nested objects:* After refinement, a component $C_i$ needs not be simply connected, i.e. $C_i$ encloses non-colliding mesh parts as shown in Figure 3.7. In this case, nested objects are detected. Nested objects are processed as follows: Edges connecting $C_i$ and a nested object are removed, and both parts are triangulated. This procedure works stable, since both components are surrounded by simple, closed polygons after local refinement.

Let us now summarize our topology adaptation system:

*Once and for all* mesh evolutions: Compute the handle database. This database can be used for all evolutions and has to be computed only once. We have computed handles for $k = 2, 3, 4$ and $\lambda_i \leq 15$.

*During* a mesh evolution:

(1) Detect self-intersections of the mesh and construct connected vertex components $C_1, \ldots, C_k$.

(2) Apply the local refinement algorithm around the components.

(3) Handle possible nested objects.

(4) Remove the vertices of $C_1, \ldots, C_k$ and its adjacent triangles and edges from the mesh.

(5) For the neighborhood data $k$ and $\lambda_1, \ldots, \lambda_k$, look up handles for a possible topology change in the database.

(6) Check each possible handle for self-intersections.

(7) Among the handles which do not produce self-intersections, take one minimizing edge length, or accept a set of triangles with least self-intersections.

After a topology change, the mesh is usually rather coarsely sampled at the location, where the mesh has been adapted. Therefore, Taubin's local smoothing method [109] is applied to these pieces.

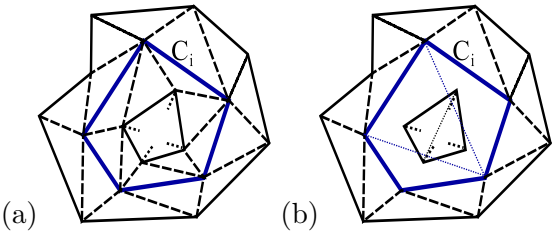



Figure 3.7: In (a), $C_i$ is not simply connected. The inside component is separated from $C_i$ as shown in (b), by removing the connecting edges. Both parts are triangulated afterwards (small dotted lines).

## 6. RESULTS AND DISCUSSION

We tested our topology change algorithm with the segmentation routine of [29] on artificial and medical images. The dark part is regarded as the object. In all examples, a small sphere is manually placed inside the voxel image, and automatically evolved towards the boundary of the object. As far as possible, we compare the experimental results to those given in [71].

- In an ultrasound dataset a cyst is segmented in Figure 3.8. The white part inside the cyst, running from front to back, stems from a biopsy needle. A segmentation is performed to determine the shape of the cyst and the position of the needle. Cyst and needle are accurately segmented.
- Figure 3.9 shows a cube with spherical cavity. Differing from the example in [71], every side of the cube contains a hole, such that the segmenting contour has genus 5.
- The left part of Figure 3.10 shows an object of genus 3, the starting ball chosen on one crossing of the four parts. Therefore, a topology change with four parts hitting at the same time is performed.
- The right part of Figure 3.10 shows a torus with 4 nested objects. Segmentation result is a torus enclosing 4 spheres.

The performance of our topology adaptation system tested on the four examples is given in table 3.1. As expected, the running times of the segmentation algorithm roughly depend linearly on the number of iterations resp. vertices. The running time for segmentation of the object of genus 3 is a bit shorter, since many vertices reach the object boundary rather early, and only a comparably small number of vertices is actually updated during an evolution step. Altogether, we obtain a speedup versus previous 3D topology adaptive segmentation routines. The cube with spherical cavity can be compared to the first example in [71]. There, only one face of the cube is penetrated by the ball, such that their object has genus 0. We obtain comparable segmentation quality in some seconds, in spite of more complex topology, more faces and more iteration cycles.

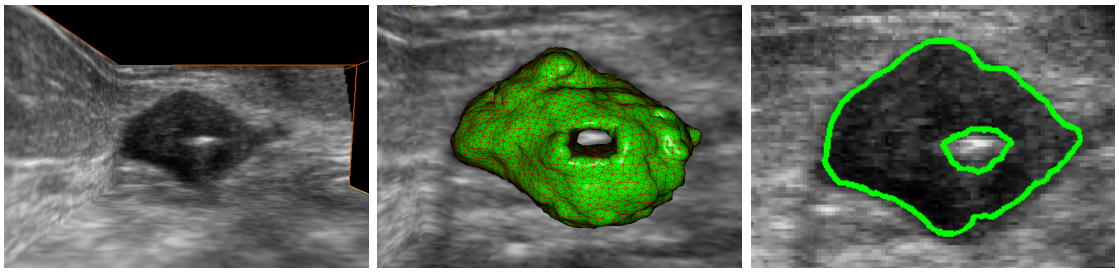


Figure 3.8: The original ultrasound image is given on the left hand side. The final segmenting mesh is shown in the middle. A projection on the y-z plane is presented on the right hand side.

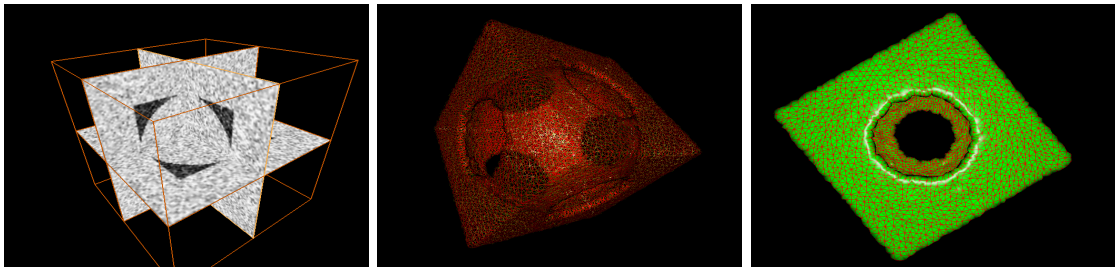


Figure 3.9: A cube with a spherical cavity and some Gaussian noise added. As a segmentation result, we obtain the mesh shown on the middle and right. In the middle diagram, only the edges are visualized.

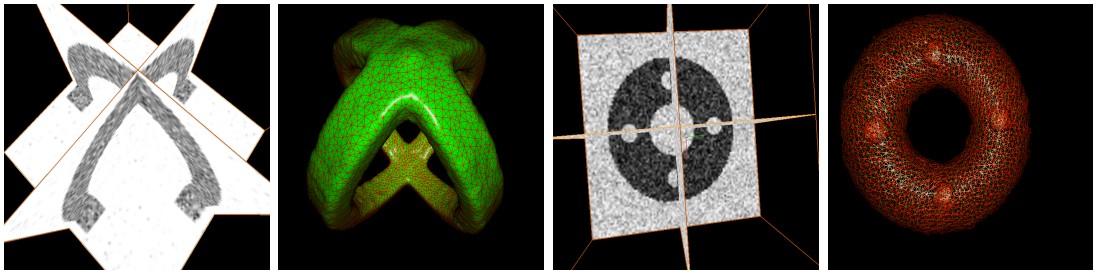


Figure 3.10: Voxel images for the last two examples, with Gaussian noise added, and the segmentation results.

| Object | Voxel size | Iterations | Number of vertices | Running time in s. |
|---|---|---|---|---|
| Cyst | $199 \times 99 \times 171$ | 133 | 8687 | 3 |
| Cube with spherical cavity | $100 \times 100 \times 100$ | 709 | 13576 | 33 |
| Object of genus 3 | $100 \times 100 \times 100$ | 677 | 9680 | 14 |
| Torus with nested objects | $100 \times 100 \times 100$ | 215 | 5454 | 6 |

Table 3.1: For each test example, the number of iterations and vertices and the running time of the segmentation algorithm is given. Tests were performed on a 3.5 GHz computer with 2 GB RAM.

## 7. CONCLUSION AND OUTLOOK

Based on a database derived from homology theory, we have introduced a very efficient novel topology adaptation system which runs independently of the evolution, does not require any reparametrizations and runs stable, even if the mesh is not regularly sampled. Based on spatial hashing, we have introduced a novel and efficient (self)-collision detection algorithm for triangular meshes, which runs in linear time and does not require complex data structures or huge memory resources. The presented examples show that accurate 3D segmentation can be performed in some seconds. As a future work, we want to combine the presented topological adaptation algorithm with a locally adaptive mesh evolution as presented in [71] to reduce the number of mesh vertices and obtain further speedup.

## ACKNOWLEDGEMENTS

The author would like to thank O. Scherzer for fruitful discussions and the anonymous reviewers for the critiques that helped to improve the paper. The work of J.A. is supported by the Austrian Science Foundation (FWF) project Y-123INF.

# 4 A combinatorial method for topology adaptations in 3D deformable models

Joint work with O. Scherzer.

Published in International Journal of Computer Vision, http://www.springerlink.com/content/69851u142747hr67.

This paper presents a topology adaptation system based on combinatorial formulas. For details on theorems 4.3.6 and 4.3.7, see Chapter 1.6.3.

External Contributions: O. Scherzer participated in discussions and revised the paper.

# A COMBINATORIAL METHOD FOR TOPOLOGY ADAPTATIONS IN 3D DEFORMABLE MODELS

Jochen Abhau [†], Otmar Scherzer [†,‡].

[†] Department of Mathematics, University of Innsbruck
Technikerstr. 21a, 6020 Innsbruck, Austria
{jochen.abhau, otmar.scherzer}@uibk.ac.at.

[‡] Radon Institute of Computational and Applied Mathematics
Altenberger Str. 69, 4040 Linz, Austria.

**Abstract.** In this paper we propose an efficient algorithm for topology adaptation of evolving surface meshes in 3D. This system has two novel features: First, a spatial hashing technique is used to detect self-colliding triangles of the evolving mesh. Secondly, for the topology adaptation itself, we use formulas which are derived from homology. In view of this the advantages of our algorithm are that it does not require global mesh re-parameterizations and the topology adaptation can be performed in a stable way via a rather coarse mesh. We apply our algorithm to segmentation of three dimensional synthetic and ultrasound data.

**Key Words.** Deformable model, triangular mesh, topology adaptation, segmentation, homology

## 1. INTRODUCTION

Since the pioneering work [116] deformable contours have been used successfully in various areas of applications, such as image processing, medical imaging, cloth modeling and game development.

It is common to differ between *explicit* and *implicit* deformable contours – that is, such are parametric and level set models respectively. The later have been introduced in [89] and since then, a number of achievements have been made both on the theoretical side [25] and on the numerical side, using additive operator splitting schemes (which are surveyed in [114]) and narrow-band methods (introduced in [5], for recent applications see also [53, 118, 57]). One advantage of implicit methods is that topology adaptations are handled automatically during the evolution process. Nevertheless explicit models are often preferred since efficient narrow-band implementations require complicated data structures and can lead to artifacts when discretizing with axes aligned bounding boxes. Having segmentation of medical images in mind, a major problem with level set methods occurs with low-contrast images. In this case, many different connected components are

segmented, whereas the user only wants to obtain the contour of one single connected object, which possibly contains some enclosed objects.

In this work we develop an explicit method which allows for topological adaptive segmentation, and we believe that it is superior to level set techniques in the above mentioned medical context. Such methods have already been subject to extensive research. To our knowledge, explicit contour models with topological adaptiveness have been considered first in [77] and [108]. There, deformable contours are represented as *tensorial spline products* [77] and sets of *dynamic particles* [108], respectively. The basic *snake model*, introduced in [83], has been complemented with topology adaptivity in [84] utilizing a supplemental Freudenthal triangulation. This triangulation is obtained by subdividing the $d$-dimensional image domain into a uniform cubic grid and further subdividing each cube into $d$ factorial simplices. With this additional simplicial structure a re-parametrization is performed periodically after a fixed finite number of iterations of the snake evolution. In each Freudenthal triangle mesh self-collisions are checked for and topology is adapted where collisions have been detected. Similar ideas have been presented in [37]. Basically, the algorithms of [84, 37] consist of three steps. First a grid is aligned on the two dimensional image domain containing the object to be segmented. Secondly, intersections of the contour edges with the grid edges are computed and stored as grid vertices. From the grid vertices new contour edges are computed, which are edges connecting the grid vertices. Thirdly, self-intersections of the re-parameterized contour are detected and the topology of the contour is adapted in all simplices composed of the grid edges. In [17] it is suggested to evolve a polygonal contour where the vertices are restricted to lie on a supplemental rectangular grid of the image domain. An advantage of this approach is that no re-parameterizations have to be performed and topology adaptations are along the lines of [84]. On the other hand, if the underlying grid is fine, small time stepping is required and thus the evolution becomes numerically expensive. In [70] a mesh transformation algorithm is proposed which discards overlapping mesh parts and performs a re-triangulation afterwards. This method only works if the mesh satisfies geometrical properties, which are controlled by a distance field evolution. According to [112, Sec. 4] distance field computations are numerically very expensive. A speed up of the algorithm of [70] has been obtained in [72, 73, 71] by relaxing the (global) geometrical constraints by local conditions. A heuristic approach to topology adaptive segmentation is chosen in [22], requiring a large number of parameters and transformation rules.

Our proposed algorithm is designed for segmentation of multiple connected surfaces in **3D** and consists of the following steps:

**4.1.1 Scheme** (Topology adaptive segmentation scheme)**.**

1. An active contour model is used to evolve a mesh until self-intersections are detected. *Detection* is performed by a spatial hashing algorithm described in Section 4. This algorithm is motivated from [111].

2. Neighboring vertices of colliding parts of the mesh $M$ are removed to get an opened mesh $M_b$ whose boundary consists of a number of simple closed polygons. Possible

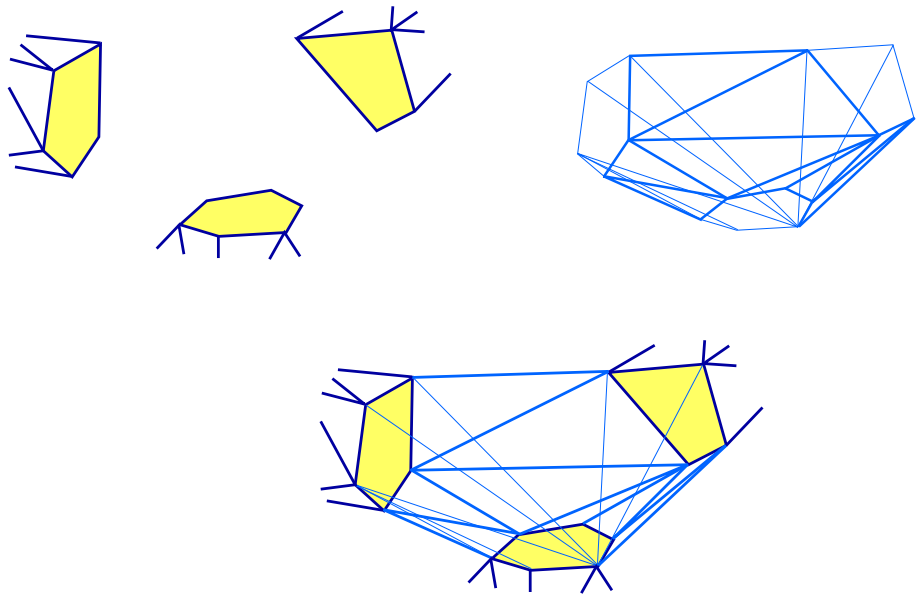


Figure 4.1: *Top Left:* Opened mesh $M_b$. *Top Right:* Handle $K$. *Bottom:* Closed mesh $M_c$.

enclosed objects are taken into account. The algorithm is described in Section 4.

3. The opened mesh $M_b$ is completed by a *handle*, which consists of a mesh $K$, that is topologically equivalent (i.e. homeomorphic) to a sphere with holes. The completed mesh $M_c$ consists of the union of $M_b$ and $K$. The topology adaption is illustrated in Figures 4.1. To make the completion algorithm efficient we use a precomputed database of topologically equivalent meshes for the handles. The database is structured by the number of connected polygons and the numbers of faces, respectively.

4. Afterwards the active contour evolution is further continued.

In this paper we focus on algorithms for detection of self intersections and topology adaptations. Active contour models are not discussed here further, we refer to [29] for a standard reference on this topic.

The *outline* of this paper is as follows: Section 4 describes the self-collision detection system. Section 4 introduces handles as the main tool to perform topological adaptations, using concepts from homology theory. Section 4 describes the complete topology adaptation system. Section 4 provides some results from segmentation of artificial and medical test images. Section 4 concludes the paper, and proofs of some theorems are given in the Appendix.

## 2. (SELF-)COLLISION DETECTION

For collision detection of the evolving surfaces we use a spatial *hashing algorithm* which is motivated from [111]. However, in comparison, our proposed algorithm has several additional features. For instance, for implementation it does not require complicated data structures and the running time is linear with respect to the number of vertices and the chosen hash table size. We are given a triangular mesh $M = (V, E, F)$ in a bounded region $\Omega \subset \mathbb{R}^3$. The proposed hashing algorithm consists of the following two steps:

**4.2.2 Scheme** (Collision Detection Algorithm by Spatial Hashing)**.**

1. For all mesh vertices $v$, eight *hash functions* $h_{i_1,i_2,i_3}(v)$ (with $i_1, i_2, i_3 \in \{0, 1\}$) are computed by a subdivision of $\Omega$ into axes aligned bounding boxes.

2. Let $i_1, i_2, i_3 \in \{0, 1\}$. For all hash values $j$ let

$$V_j^{i_1,i_2,i_3} = \{\text{vertices with } h_{i_1,i_2,i_3}(v) = j\},$$

the sets of vertices with hash value $j$. In this step it is checked whether triangles containing vertices of $V_j^{i_1,i_2,i_3}$ intersect.

In the following we present some details of the spatial hashing algorithm. In the first step, for a definition of the hash functions, we use large prime numbers $p_i$, $i = 1, 2, 3$, and choose a hash table size htblSize. Moreover, we denote by the real parameter $l$ the size of the axes aligned bounding boxes (see Figure 4.2). By $\lfloor a \rfloor$ we denote the greatest integer smaller than $a$. For $a > 0$ and $i \in \{0, 1\}$ let

$$r(a, i) = \begin{cases} \left\lfloor \frac{a}{l} \right\rfloor l & \text{if } i = 0 \\ \left\lfloor \frac{a}{l} + \frac{1}{2} \right\rfloor l & \text{if } i = 1 \end{cases} \tag{4.1}$$

For $i_1, i_2, i_3 \in \{0, 1\}$ we define hash functions:

$$h_{i_1,i_2,i_3}(v) = r\left(\frac{v_x}{l}, i_1\right) p_1 + r\left(\frac{v_y}{l}, i_2\right) p_2 + r\left(\frac{v_z}{l}, i_3\right) p_3 \bmod (\text{htblSize} + 1) \tag{4.2}$$

**4.2.3 Theorem.** *If $\|P - Q\| \leq \frac{l}{2}$, then at least for one of the eight tripels $(i_1, i_2, i_3) \in \{0, 1\}^3$, we have*

$$h_{i_1,i_2,i_3}(P) = h_{i_1,i_2,i_3}(Q) \tag{4.3}$$

*Proof.* Since $\|P - Q\| \leq \frac{l}{2}$ we have $|P_j - Q_j| \leq \frac{l}{2}$ for all $j = 1, 2, 3$. For each $j$, one of the following two statements holds:

(1) there exists $k \in \mathbb{N}$ such that $P_j, Q_j \in [kl, (k + 1)l]$

(2) there exists $k \in \mathbb{N}$ such that $P_j, Q_j \in [(k - \frac{l}{2}), (k + \frac{1}{2})l]$

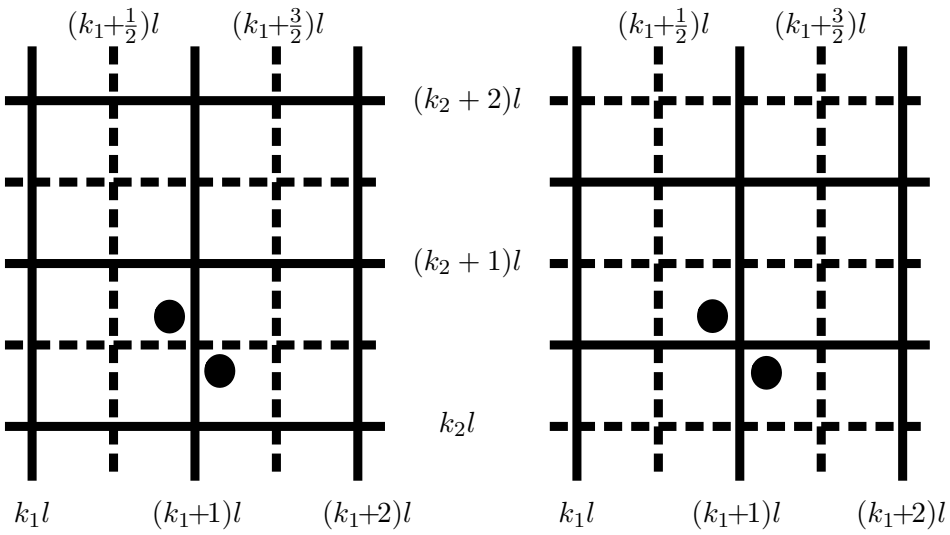
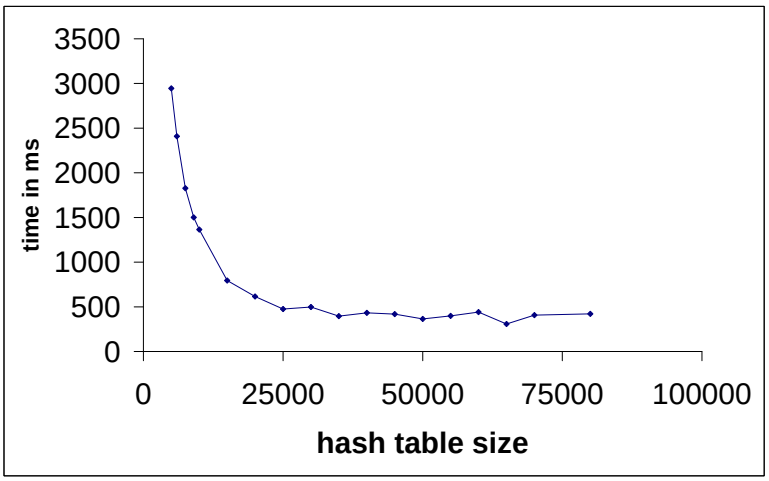



Figure 4.2: A 2D-projection of the hash function construction. In 2D, four systems of axes aligned bounding boxes cover $\Omega$. Two systems are shown on the left hand side, and two systems are shown on the right hand side, respectively. Points closer than $l/2$ to each other are contained in at least one common square. The two marked points are detected by the dashed system on the right hand side. In 3D, eight hash functions are used.



If (1) holds, we choose $i_j = 0$, otherwise we choose $i_j = 1$. Then for every $j$, we have $r\left(\frac{P_j}{l}, i_j\right) = r\left(\frac{Q_j}{l}, i_j\right)$ and therefore equation (4.3) holds. □

We use vertices with Euclidean distance smaller than $l/2$ as indicators for intersecting triangles.

For an optimal choice of the box size $l$ we use Theorem 4.2.4, below. For a proof of the theorem we refer to the Appendix.

**4.2.4 Theorem.** *Assume that the length of every edge of a mesh $M = (V, E, F)$ is bounded by $s$. Moreover, we assume that the triangles $T = (T_1, T_2, T_3)$ and $S = (S_1, S_2, S_3)$ of the mesh intersect. Then there exist $i, j$ such that*

$$\|T_i - S_j\| < \sqrt{\frac{2}{3}}s.$$

*Here and in the following we identify the triangle with the triple of edge points.*

According to the theorem we choose $l > 2\sqrt{2/3}s$, because this choice guarantees, that at least two vertices of intersecting mesh triangles are mapped to the same hash key, and thus in the sequel an intersection test is performed. For our applications we have chosen the hash table size to be twice the number of mesh vertices. This choice is based on numerical experiments with meshes of approximately 10k vertices and 20k triangles, respectively. The hash table sizes have been varied (see Figure 4.2). From this table we see that the running time for the collision detection is monotonously decreasing in hash table sizes smaller than twice the number of mesh vertices, and remains nearly constant for greater hash table sizes. Therefore, from a point of memory usage the suggested choice appears to be optimal. In the second step we iterate through the hash keys and check for each pair of non-neighboring vertices with the same hash key if they are contained in triangles which intersect. This is done with a fast triangle-triangle intersection test, see [86].

## 3. THE HANDLE DATABASE

In this section we show how to generate a data base of simplices which allows for completion of an opened mesh $M_b$ to a closed mesh $M_c$. A set of such simplices will be called a *handle*. Here we make use of concepts from topology (see e.g. [56, 82, 96] for background material). We are mainly concerned with combinatorics and use simplicial complexes and subcomplexes in a coordinate free abstract sense as in [96, p.141]. We recall that *abstract simplicial complexes* are not necessarily defined by coordinates in Euclidean space, but only by abstract data like integers. Therefore the geometric realization $|M|$ of an abstract simplicial complex $M$ (see [96, p.142]) is only defined up to homeomorphism.

To start with, we make a basic definition of handles:

**4.3.5 Definition.** We call an abstract simplicial complex $C = (C_0, C_1, C_2)$ a *cap*, if there exist $m \in \mathbb{N}$, $a_0, \ldots, a_m \in \mathbb{Z}$, $a_0 < \ldots < a_m$ such that:

$$\begin{aligned} C_0 &= \{a_0, \ldots, a_m\} \ , \\ C_1 &= \{\{a_0, a_1\}, \ldots, \{a_0, a_m\}, \{a_1, a_2\}, \{a_2, a_3\}, \ldots, \{a_{m-1}, a_m\}, \{a_m, a_1\}\} \ , \\ C_2 &= \{\{a_0, a_1, a_2\}, \{a_0, a_2, a_3\}, \ldots, \{a_0, a_{m-1}, a_m\}, \{a_0, a_m, a_1\}\} \ . \end{aligned}$$

The orientation of the complex is given by $(a_0, a_1, a_2)$, $(a_0, a_2, a_3)$, ..., $(a_0, a_{m-1}, a_m), (a_0, a_m, a_1)$. $a_0$ is called the *vertex center*.

Let $C^1, \ldots, C^k$ be caps. An abstract simplicial complex $K = (K_0, K_1, K_2)$ is called *handle* if the geometric realization $|\bigcup_{i=1}^k C^i \cup K|$ of

$$\bigcup_{i=1}^k C^i \cup K := \left( \bigcup_{i=1}^k C_0^i \cup K_0, \bigcup_{i=1}^k C_1^i \cup K_1, \bigcup_{i=1}^k C_2^i \cup K_2 \right)$$

is homeomorphic to a 2-sphere (in signs $|\bigcup_{i=1}^k C^i \cup K| \approx \mathbb{S}^2$) and for all $j \in \{1, \ldots, k\}$, the inclusion

$$C^j \hookrightarrow \bigcup_{i=1}^k C^i \cup K$$

is orientation preserving. We recall that homeomorphisms are defined to respect topological properties. A cap is visualized in Figure 4.3.

The following theorem characterizes topological properties of a handle and states that the a mesh $M$ after opening at $k$ locations and closing by a handle (this is the mesh $M_c$) constitutes a surface which has $k-1$ tunnels more than $M$, that is the genus is increased by $k-1$.

**4.3.6 Theorem.** *Let $M = (V, E, F)$ be an abstract 2-dimensional simplicial complex, such that $|M|$ is an orientable, connected surface without boundary. Furthermore, let $M_b$ be the simplicial subcomplex of $M$ obtained by removing $k$ connected components from $M$ such that $|M_b|$ is a surface with boundary consisting of $k$ simple closed polygons (see also Figure 4.1, top left). Let $C^1, \ldots, C^k$ be caps such that its boundaries consist of the boundary polygons of $M_b$ (see also Figure 4.3). Moreover, let $K$ be a handle for $C^1, \ldots, C^k$, so that $|\bigcup_{i=1}^k C^i \cup K| \approx \mathbb{S}^2$. Then*

$$|M_b \cup K| \approx T_{k-1+g(M)}, \tag{4.4}$$

*where $T_{k-1+g(M)}$ is the closed, orientable surface of genus $k-1+g(M)$ and $g(M)$ is the genus of $M$.*

This theorem can be proven by standard methods of algebraic topology.

Topological equivalent 2-spheres can be characterized by simplicial homology (see e.g. [96, p.144] for background on this topic).

**4.3.7 Theorem.** *Let $M = (V, E, F)$ be an abstract 2-dimensional simplicial complex such that every edge $e \in E$ is a face of some $f \in F$. If the homology conditions*

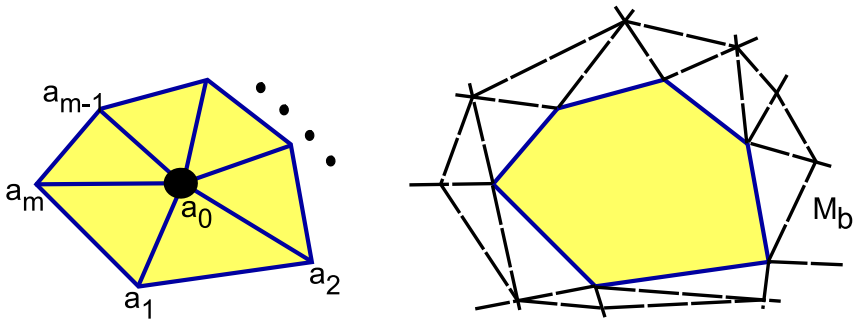



Figure 4.3: Left: A cap. Right: A simple closed boundary polygon of $M_b$.

- $H_0(M) \cong \mathbb{Z}$
- $H_1(M) = 0$
- $H_2(M) \cong \mathbb{Z}$, *generated by* $\sum_{f \in F} \epsilon_f f$ *with* $\epsilon_f \in \{-1, 1\}$

*hold, then the geometric realization of* $M$ *is homeomorphic to* $\mathbb{S}^2$.

This theorem is similar to the well-known *Whitehead Theorem* (see [56, p.346]) which states that spaces with isomorphic homotopy groups are homotopy equivalent. Theorem 4.3.7 can be proven again by standard methods of algebraic topology.

Based on these basic definitions and Theorem 4.3.7 we can present an algorithm for computation of handles which contain given simple closed polygons as boundary polygons. With each simple closed polygon we can associate a cap by adding a center vertex and connecting the center with the vertices by edges. Therefore, we concentrate now on computation of handles given disjoint caps.

Let us assume that we have given $k$ disjoint caps $C^i$, $i = 1, \ldots, k$ with numbers of vertices $m_i$, $i = 1, \ldots, k$, respectively.

We use the notation $m = \sum_{i=1}^{k} m_i$ and denote by $v_c, e_c, f_c$ the numbers of vertices, edges, and faces of the mesh $\bigcup_{i=1}^{k} C^i \cup K$. Let us assume that the geometric realization of $\bigcup_{i=1}^{k} C^i \cup K$ is homeomorphic to a sphere – that is the case if $K$ is a handle. Then, from the Euler formula (see [96, p.146]) we know that $v_c - e_c + f_c = 2$. Moreover, by induction on the number of triangles, we can show that $2e_c = 3f_c$, and therefore $f_c = 2v_c - 4$. If $k$ caps are connected by a handle, the number of vertices of the arising sphere is $v_c = m + k$, and therefore

$$f_c = 2m + 2k - 4.$$

Since the caps $C^i$ contain $m$ faces altogether, $m + 2k - 4$ faces have to be added to the faces of the set $\{C^i : i = 1, \ldots, k\}$ to obtain a sphere. We differ between two different kinds of faces to be added (see Figure 4.4):

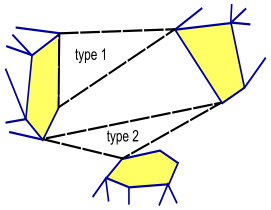



Figure 4.4: Different types of faces.

- When a face to be added has two vertices in common with some $C^i, i = 1, \ldots, k$, then it is called of Type 1.
- Else it is called of Type 2. That is the case if the face has at most one vertex in common with every $C^i$.

There exist $m$ faces of Type 1 and $2k - 4$ faces of Type 2.

Based on these considerations we are able to generate the handle database, which associates tuples consisting of the number of connected polygons $k$ and the numbers of vertices $m_i$, $i = 1, \ldots, k$, a set of handles, respectively. Without loss of generality we always assume in the sequel that $m_{i+1} \geq m_i$, $i = 1, \ldots, k-1$. For given $k$ and $m_i$, $i = 1, \ldots, k$, a handle of the database is determined as follows:

1. For each boundary edge of a cap choose a vertex in a different cap. The face determined by the edge and the chosen vertex is one of the $m$ triangles of Type 1.
2. Locate two edges which share a common vertex such that all three vertices are contained in different caps. This determines the $2k - 4$ triangles of Type 2.
3. Check if the abstract simplicial complex made of the caps and the added faces is a sphere, that is, it satisfies the conditions of Theorem 4.3.7. Computationally, one can check the homology criterion using the PARI software [90].

Given caps $C^i$, $i = 1, \ldots, k$ with $m_i$ vertices, respectively, it is useful for our purposes to associate a sequential enumeration to the vertices. To this end we use the notation $\mu_l = \sum_{i=1}^{l} m_i$. Vertices between $\mu_{l-1} + 1$ and $\mu_l$ (where we set $\mu_0 := 0$) correspond to the vertices in the cap $C^l$.

For $i \in \{1, \ldots, \mu_k\}$, we set

$$i \oplus 1 = \begin{cases} \mu_{j-1} + 1 & \text{if } i = \mu_j \text{ for some } j \in \{1, \ldots, k\} \\ i + 1 & \text{otherwise.} \end{cases}$$

Therefore, $i \oplus 1$ is the subsequent vertex of $i$ in the cap $C^i$.

**4.3.8 Example.** In this example we calculate the number of different elements of the handle database for some test cases of small $k$.

| $m_1$ | $m_2$ | $m_3$ | $m_4$ | handles | generators |
|---|---|---|---|---|---|
| $k = 3$: | | | | | |
| 3 | 3 | 4 | - | 72 | 1 |
| 3 | 4 | 5 | - | 120 | 2 |
| 4 | 4 | 6 | - | 192 | 2 |
| $k = 4$: | | | | | |
| 3 | 3 | 4 | 4 | 576 | 1 |

Table 4.1: The numbers of possible handles in the database, and the number of generators taking into account group actions.

$k = 2$: Because we have $2k - 4 = 0$, only faces of Type 1 occur. For a function

$$f : \{1, \ldots, m_1\} \to \{m_1 + 1, \ldots, m_1 + m_2\} \ ,$$

which we assume to be monotonously decreasing and surjective we define

$$g : \{m_1 + 1, \ldots, m_1 + m_2\} \to \{1, \ldots, m_1\} \, , \\ j \mapsto \max \{i : f(i) = j\} \, .$$

Note that $f$ maps vertices of the first cap onto vertices of the second and $g$ is a right inverse. These two functions define a handle with the face set

$$\{(1, 2, f(1)), \ldots, (m_1 - 1, m_1, f(m_1 - 1)), (m_1, 1, f(m_1)), \\ (m_1 + 1, m_1 + 2, g(m_1 + 1)), \ldots, \\ (m_1 + m_2 - 1, m_1 + m_2, g(m_1 + m_2 - 1)), \\ (m_1 + m_2, m_1 + 1, g(m_1 + m_2))\} \ .$$

For $k = 3, 4$ and some tuples of edge numbers the numbers of possible handles have been summarized in Table 4.1.

We consider symmetric group actions on the set of handles $\mathcal{K} = \mathcal{K}(k; m_1, \ldots, m_k)$ for $k$ caps of sizes $m_1, \ldots, m_k$, respectively. For the theory of group actions, see [74, p.25]. By a group action on a set $S$, the set $S$ is partitioned into disjoint orbits $S_1, \ldots, S_n$ (see [74, p.28]), and a set of generators of $S$ is a choice of elements $s_i \in S_i$, $i = 1 \ldots, n$. We identify generators of $\mathcal{K}$ which completely determine $\mathcal{K}$ modulo group actions.

**Rotations:** The topology of the caps remains unchanged if a rotation of the vertex

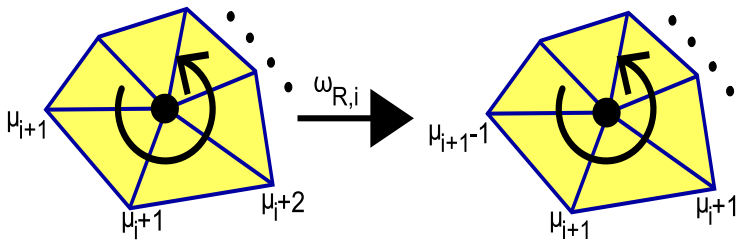



Figure 4.5: A rotation.

numbers of a cap is performed. More precisely, let

$$\omega_{R,i}(l) = \begin{cases} l \oplus 1 & \mu_i < l \leq \mu_{i+1} \\ l & \text{otherwise,} \end{cases} ,$$
$$\Omega_R = \Omega_{R,1} \times \ldots \times \Omega_{R,k} ,$$
$$\Omega_{R,i} = \text{ subgroup of } \Sigma_{\mu_k} \text{ generated by } \omega_{R,i} ,$$
$$\Sigma_{\mu_k} \text{denotes the symmetric group on the set } \{1, \ldots, \mu_k\} .$$

We obtain a group action of $\Omega_R$ on $\mathcal{K}$ by applying $\Omega_R$ to every vertex of every simplex of a handle $K$. Rotations are illustrated in Figure 4.5.

**Exchanges:** The order of two caps, consisting of the same number of vertices, can be exchanged. If

$$\omega_{E,i,j}(l) = \begin{cases} l - \mu_j + \mu_i & \mu_j < l < \mu_{j+1} \\ l + \mu_j - \mu_i & \mu_i < l < \mu_{i+1} \\ l & \text{otherwise} \end{cases}$$

and $\Omega_{E,i,j} = \{\text{id}, \omega_{E,i,j}\}$ the group of exchanges of cap $i$ and cap $j$, then

$$\Omega_E = \prod_{i<j, m_i = m_j} \Omega_{E,i,j}$$

operates on $\mathcal{K}$. Exchanges are illustrated in Figure 4.6.

An easy computation shows that the two operations commute, i.e.

$$\omega\tau(K) = \tau\omega(K) \text{ for } \omega \in \Omega_R, \tau \in \Omega_E \text{ and } K \in \mathcal{K}.$$

As a consequence, we can apply rotations and exchanges in an arbitrary order to a handle. With these operations, only very few elements are required to generate elements of $\mathcal{K}$. This is illustrated by comparing the last two columns of Table 4.1.

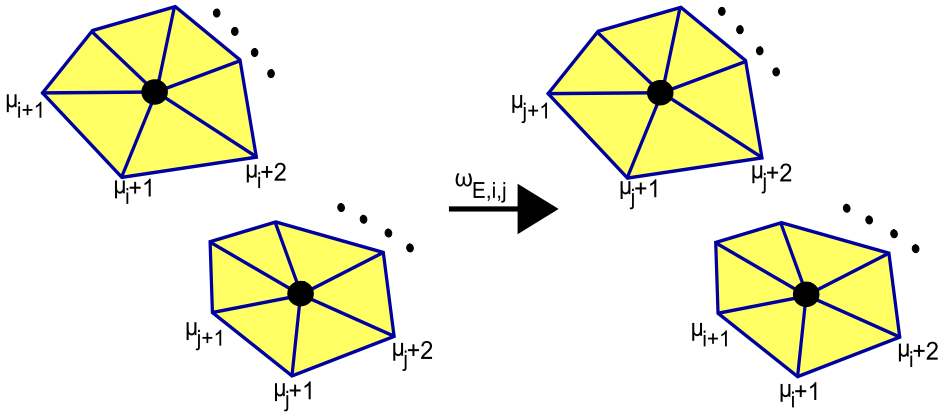



Figure 4.6: An exchange

# 4. IMPLEMENTATION OF THE TOPOLOGY ADAPTATION

For implementation of the topology adaptation algorithm (compare Scheme 4.1.1), we use the handle database as main tool. Moreover, further routines for mesh opening and splitting into components are implemented, which we describe below.

We are given an initial triangular mesh $M_0 = (V_0, E_0, F_0)$. The initial mesh is assumed to be free of self-intersections and without boundary, but several components of arbitrary genus are allowed. An iterative evolution is performed on the mesh. (Self-)intersections of the evolving contour are detected by the spatial hashing algorithm of Section 4. This algorithm computes vertices with the same hash index which belong to intersecting triangles.

**Mesh Opening.**
The opening of the mesh is performed in such a way that the boundary of the opened mesh $M_b$ consists of simple closed polygons. For this purpose vertices of colliding mesh parts are grouped into disjoint connected sets $\Lambda^1, \ldots, \Lambda^k$ such that for $i \neq j$, two arbitrary vertices $v \in \Lambda^i$, $w \in \Lambda^j$ have no common neighbor. The set $N_i$ of all neighboring vertices of vertices in $\Lambda^i$ without $\Lambda^i$ is a neighborhood of $\Lambda^i$, which consists of connected components $N_i^0, \ldots, N_i^{l_i}$. We assume that the neighborhoods $N_i^j$ are pairwise disjoint and that its edges form a simple, closed polygon, otherwise the following neighborhood refinement routine is used: We insert new vertices on the bisectors of edges between vertices of $\Lambda^i$ and of $N_i^j$, and connect these vertices by edges as shown in Figure 4.7. Arising quadrilaterals are triangulated. As a result, the edges connecting the bisectors form a simple closed polygon around $\Lambda^i$, and their neighborhoods are pairwise disjoint. The routine is illustrated in Figure 4.7(a)-(d).

**Splitting.**

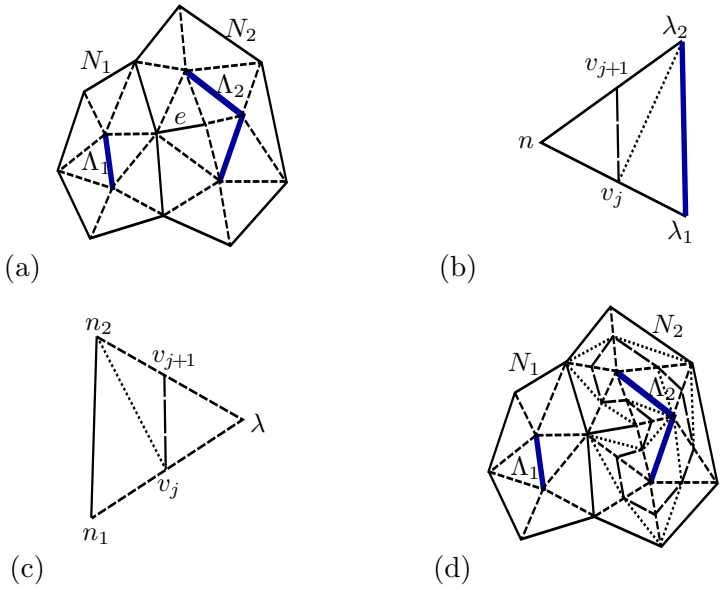



Figure 4.7: In (a), $\Lambda_1$ consists of two and $\Lambda_2$ of three vertices connected by a bold line. $N_1$ and $N_2$ have three common neighbor vertices, and $N_2$ forms no simple closed polygon because of edge $e$. The iterative refinement algorithm is demonstrated in (b) and (c). Bisector vertices $v_j$ are inserted as well as edges between them (dashed line), and the arising quadrilaterals are triangulated (dotted line). The result of the refinement algorithm is shown in (d).

In general, the neighborhood $N_i$ consists of several components $N_i^0, \ldots, N_i^{l_i}$. One component, say $N_i^0$, encloses $\Lambda^i$, and the other components are enclosed by $\Lambda^i$, see Figure 4.8. The outside component $N_i^0$ can be computed from the orientation of the mesh. Components $N_i^1, \ldots, N_i^{l_i}$ belong to enclosed parts of the mesh. There are two different kinds of enclosed object neighborhood components $N_i^j$ for $j \geq 1$ and we propose two different procedures:

- If $N_i^j$ contains a triangle, i.e. there exist $v_1 \in N_i^j$, $v_2, v_3 \in V \backslash \Lambda^i$ neighbors of $v_1$, s.t. $(v_1, v_2, v_3) \in F$. Then the neighborhood refinement routine is applied such that the boundary of $\Lambda^i$ towards $N_i^j$ as well as the boundary of $N_i^j$ towards $\Lambda^i$ are Jordan polygons. The connections between the two polygons are discarded. For each polygon, the barycenter of the vertices is inserted, and connected to the polygon. Thus, the mesh is split into two separate components.
- If $N_i^j$ contains no triangle, no real enclosed object has been detected, and $N_i^j$ is added to $\Lambda^i$.

**Deleting components.**
After discussing possible splittings, we can assume that the neighborhood $N_i$ of a set $C_i$

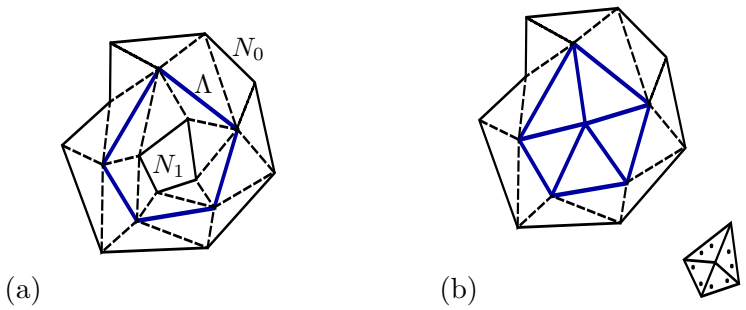



Figure 4.8: In (a), the neighborhood of $\Lambda$ consists of two components. $N_0$ is the component outside $C$, $N_1$ is an enclosed component. In (b), the edges connecting $\Lambda$ and $N_1$ are deleted, barycenters and connecting edges inserted, and two components arise.

is connected and a simple, closed polygon. In case that $C_i$ does not collide with another set $C_j$, we consider the following possible procedures:

- The mesh component containing $C_i$ and $N_i$ consists of no more vertices than $C_i \cup N_i$. Then this component is deleted.
- Otherwise $C_i$ is rather coarse and smoothed by Taubin's method [109].

Note that every decrease in genus is obtained by vanishing components.

**Inserting handles from the database.**
From now on, assume that $C_1, \ldots, C_k$ are overlapping, with $k \geq 2$. In this situation, we discard the vertices of $C_1, \ldots, C_k$ and its adjacent mesh elements. The neighborhoods $N_1, \ldots, N_k$ have to be reconnected by inserting edges and faces between them. In this situation, we use the handle database. Every handle gives a possible connection of the neighborhood polygons. The only thing to check is if the triangles given by the formulas self-intersect. This test is performed by the triangle-triangle intersection test of [86]. Then among the possible handles the one with shortest edge length is taken. We believe that this choice is the most natural one and gives suitable meshes for further evolution.

We implemented the topology adaption system for an active contour evolution, where only outward movement of the vertices is possible. Therefore, the volume bounded by the mesh is monotonously increasing, and no infinite loops (consisting of handle attachment and splitting) are possible. Furthermore, mesh splittings only arise, when an enclosed object has been detected. On the other hand, if mesh shrinking is also allowed, the topology adaptation system works as well, but infinite loops are possible. This arises naturally in the simple case where the object to segment consists of two pieces with small overlap, like $[-1, \epsilon]^3 \cup [-\epsilon, 1]^3$ for some small $\epsilon > 0$ or also $\epsilon = 0$. In such a case with noisy data, the user has to choose the parameters of his active contour model (like edge length bounds) appropriately to ensure convergence of the segmentation algorithm.

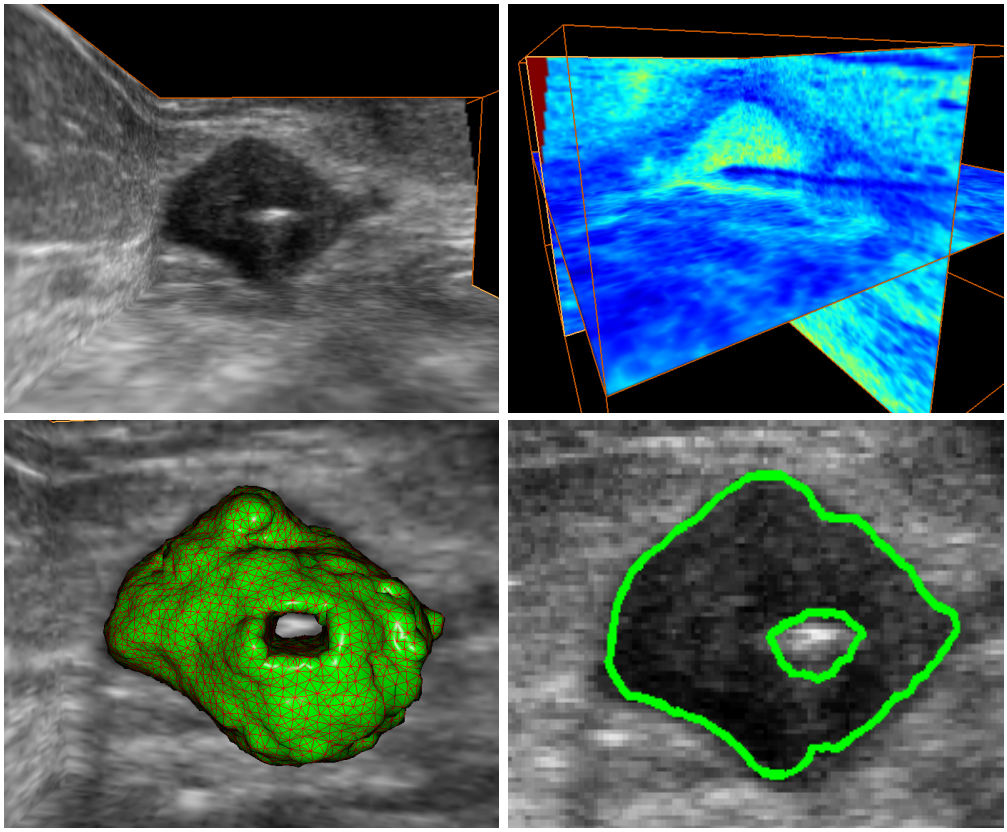


Figure 4.9: The upper left image shows the original ultrasound data. The upper right image shows the biopsy needle in greater detail, using a contrast enhancing colormap. The final segmenting mesh is shown at lower left. A projection on the y-z plane is presented at lower right.

# 5. RESULTS AND DISCUSSION

We tested our algorithm for topology adaptation in connection with the active contour algorithm published in [3] on artificial and medical test images. In both cases the dark part is regarded as the object to be segmented. To initialize the segmentation algorithm a small sphere is manually placed inside the dark part of the voxel image. The evolving surface moves towards the boundary of the object.

As far as possible we compare the numerical results to those given in [72].

- An ultrasound image of a cyst is segmented. The white part inside the cyst, running from front to back, stems from a biopsy needle, see Figure 4.9. The segmentation is used to determine the shape of the cyst and the position of the biopsy needle. As the projection to the y-z plane shows the cyst and the needle are accurately segmented, also in regions where the topology of the mesh has been adapted during

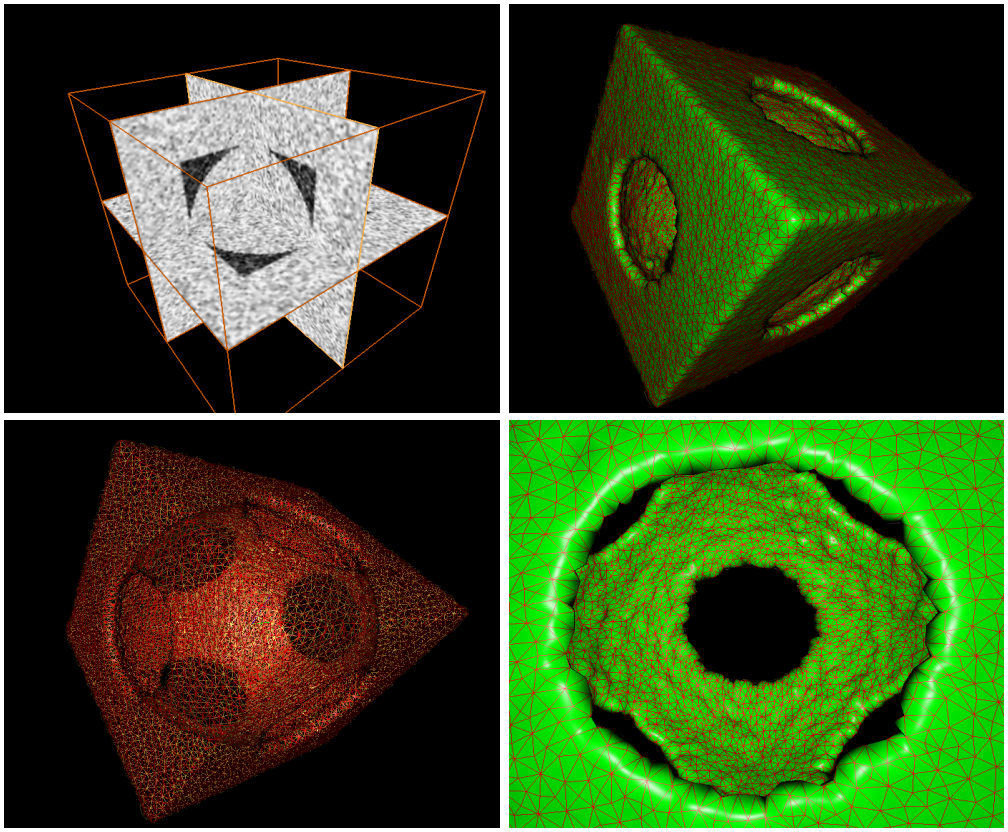


Figure 4.10: A cube with a spherical cavity and some Gaussian noise added is shown at upper right. As a segmentation result we obtain the mesh shown in the remaining images. At lower left, only the edges are visualized.

| Object | Voxel size | Iterations | Vertices | Sec. |
|---|---|---|---|---|
| Cyst | $199 \times 99 \times 171$ | 133 | 8687 | 9 |
| Cube | $100 \times 100 \times 100$ | 709 | 13576 | 65 |
| Genus 3 | $100 \times 100 \times 100$ | 677 | 9680 | 28 |
| Torus | $100 \times 100 \times 100$ | 215 | 5454 | 8 |

Table 4.2: For each test example, the number of iterations and vertices and the running time of the segmentation algorithm is given. Tests were performed on a 3.5 GHz computer with 2 GB RAM.

the evolution of contour.

- The next example concerns a computer generated voxel image of a cube with a spherical cavity (see Figure 4.10). Different from the example in [72], every side of

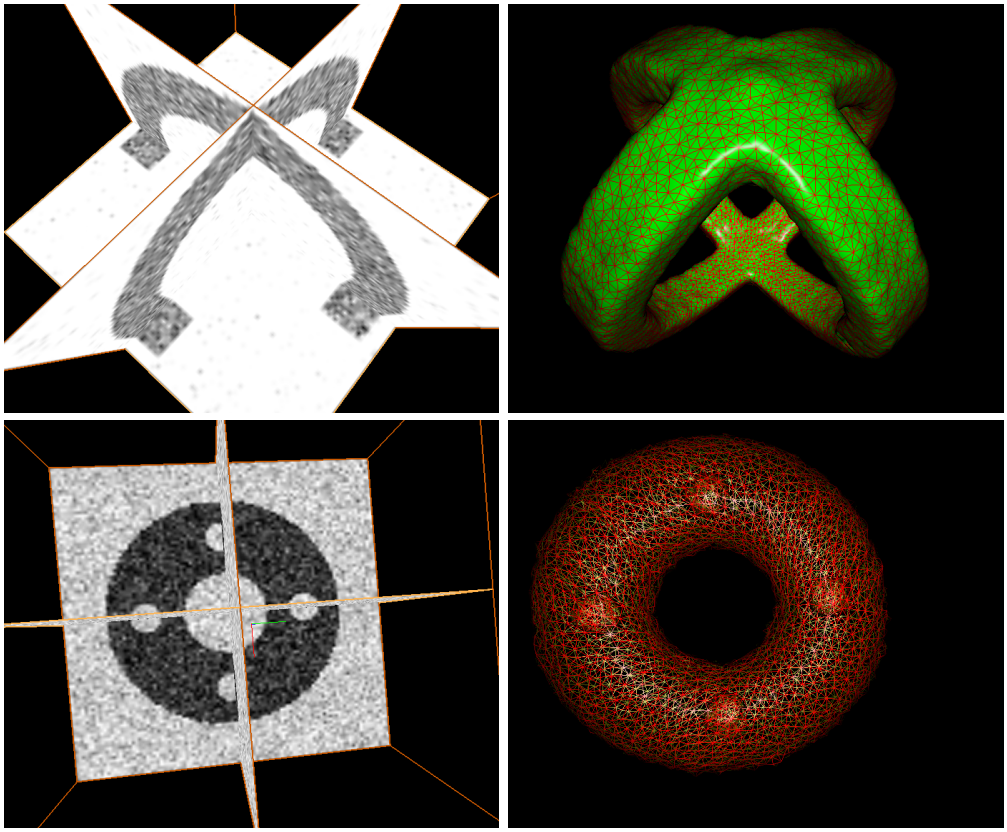


Figure 4.11: Voxel images for the last two examples and the segmentation results.

the cube contains a hole such that the segmenting contour of the object has genus 5.

- The next example shows an object of genus 3, the starting ball chosen on one crossing of the four parts. Therefore, a topology change with four parts hitting at the same iteration step is performed. The result is shown in the top part of Figure 4.11.
- The last example shows a torus with 4 enclosed objects. As segmentation result a torus enclosing 4 spheres is obtained, see the bottom part of Figure 4.11.

The performance of our topology change algorithm tested on the four examples is summarized in Table 4.2.

The numerical experiments demonstrate the robustness and efficiency of the topology completion algorithm. Its basic components, consisting of the handle database and triangle-triangle intersection tests, do not assume global mesh restrictions. On the other hand, the collision detection system requires globally bounded edge lengths, since otherwise there is no efficient choice of the box sizes. However, this is no substantial drawback,

since mesh regularity assumptions are usually inferred for a reliable computation of the force terms directing the evolution. As expected, the running times of the segmentation algorithm roughly depend linearly on the number of iterations, respectively vertices. The running time for segmentation of the object of genus 3 is a bit shorter than expected, because many vertices reach the object boundary rather early, and only a comparably small number of vertices is actually updated during an evolution step. As a consequence, we obtain a speedup versus previous 3D topology adaptive segmentation routines. The cube with spherical cavity can be compared to the first example in [72]. There, only one face of the cube is penetrated by the ball, such that their object to recover has genus 0. Due to possibly different computing modalities, it is hardly possible to compare pure computation times to [72], but our result seems to be very promising.

# 6. CONCLUSION AND OUTLOOK

We introduce a very efficient novel topology completion system which runs independently of the evolution, does not require any reparameterizations and runs stable, even if the mesh is not regularly sampled. We introduce a novel and efficient collision and self-collision detection algorithm for triangular meshes, which runs in linear time and does not require complex data structures or huge memory resources. The system is designed for interactive applications.

Due to the topological completion formulas obtained by the homology criterion, we were able to develop a very robust topological completion system, working with arbitrary mesh deformation algorithms. Since our (self-)collision detection algorithm works in linear expected time, the system is also very efficient resulting in significantly reduced computation times. For numerical experiments, we used a standard balloon model, thus losing overall efficiency for segmentation a bit. As a future work, it seems to be interesting to combine the presented topological completion algorithm with a locally adaptive mesh evolution as presented in [72, 71].

## Acknowledgement

This work has been supported by the Austrian Science Fund (FWF) within the national research networks Industrial Geometry, project 9203-N12, and Photoacoustic Imaging in Biology and Medicine, project S10505-N20. We thank GE - Medical Systems, Kretztechnik, for providing the voxel image of the cyst and Tobias Riser for his QT-Viewer, with which the example pictures of the cyst and the torus were visualized. The QT viewer has been developed within the TWF project *Parallelisierte Datenauswertung am HPC, GZ:UNI-0404/460*. We also want to thank the referees for their very useful comments.

## Appendix

For a triangle $T$ in $\mathbb{R}^3$ given by its vertices $T_1, T_2, T_3$ and a point $P$ in $\mathbb{R}^3$ let

$$d(P,T) := \min\left\{\|P-T_1\|, \|P-T_2\|, \|P-T_3\|\right\}.$$

We use the notation $T = (T_1, T_2, T_3)$ and denote by $\mathrm{pr}_T(P)$ the orthogonal projection of $P$ in the plane spanned by $T$ - which is of course only well-defined if the triangle does not degenerate.

**4.5.9 Lemma.** *Assume that $T = (T_1, T_2, T_3)$ is a triangle in $\mathbb{R}^3$, and $P \in \mathbb{R}^3$. Then*

$$d(P,T) = \sqrt{\|P - pr_T(P)\|^2 + d(pr_T(P), T)^2}.$$

*Proof.* The situation is illustrated in Figure 4.12(a). Looking at the Voronoi diagram of the three points $T_1$, $T_2$, $T_3 \in \mathbb{R}^3$, we deduce that for some $i = 1,2,3$, $\|P - T_i\| \leq \|P - T_j\|$ for all $j = 1,2,3$ if and only if

$$\|\mathrm{pr}_T(P) - T_i\| \leq \|\mathrm{pr}_T(P) - T_j\| \text{ for all } j = 1,2,3. \tag{4.5}$$

Therefore, if $d(P,T) = \|P - T_i\|$ for some $i$, we have

$$\begin{aligned} d(P,T) = \|P - T_i\| &= \sqrt{\|P - \mathrm{pr}_T(P)\|^2 + \|\mathrm{pr}_T(P) - T_i\|^2} \\ &\underset{(4.5)}{=} \sqrt{\|P - \mathrm{pr}_T(P)\|^2 + d(\mathrm{pr}_T(P), T)^2}. \end{aligned}$$

□

Now we can give a proof of Theorem 4.2.4:

**Proof of Theorem 4.2.4.**
Without loss of generality we can assume that every edge of both triangles $S$ and $T$ has maximal edge length $s$, i.e. both triangles are equilateral. Moreover, we assume that an edge $e$ of $S$ intersects $T$ (otherwise we interchange the role of $S$ and $T$), and we denote the intersection point by $Q$.

Let $P$ be an endpoint of $e$ which fulfils $\|P - Q\| \leq \frac{1}{2}s$. Denote by $\mathrm{bar}_T$ the barycenter of $T$. (see Figure 4.12(a)). We consider two cases concerning the position of $\mathrm{pr}_T(P)$:

- $d(\mathrm{pr}_T(P), T) \leq d(\mathrm{bar}_T, T)$: In this case, we have that

  $$\|P - \mathrm{pr}_T(P)\| \leq \|P - Q\| \leq \frac{s}{2}$$

  and

  $$d(\mathrm{pr}_T(P), T) \leq d(\mathrm{bar}_T, T) = \frac{1}{3}\sqrt{3}s.$$

  Therefore, using Lemma 4.5.9 we deduce that

  $$d(P,T) = \sqrt{\|P - \mathrm{pr}_T(P)\|^2 + d(\mathrm{pr}_T(P), T)^2} \leq \sqrt{\frac{1}{4} + \frac{1}{3}}s \leq \sqrt{\frac{2}{3}}s.$$

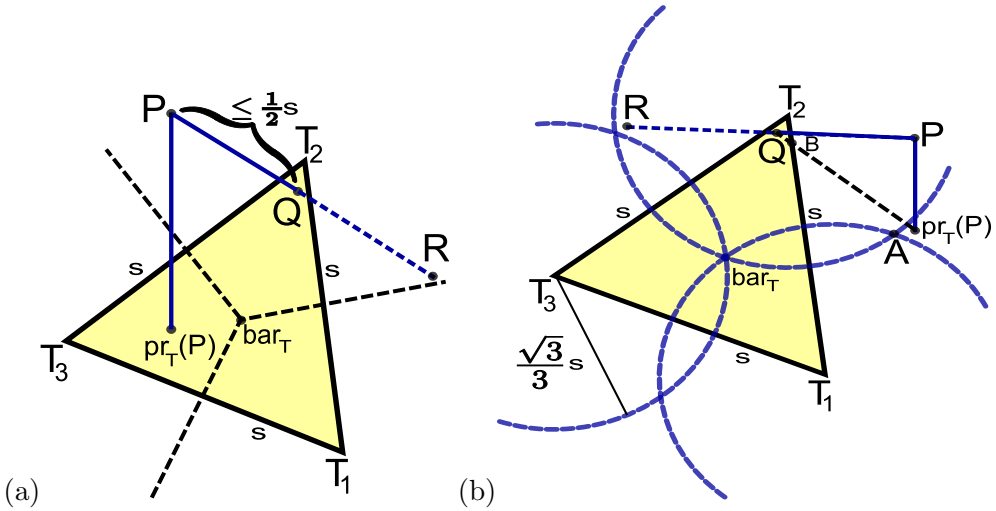



Figure 4.12: In (a), edge $\overline{PR}$ intersects triangle $(T_1, T_2, T_3)$ in $Q$. In (b), the projection $\mathrm{pr}_T(P)$ of $P$ onto the plane spanned by $T$ lies outside the three circles.

- $d(\mathrm{pr}_T(P), T) > d(\mathrm{bar}_T, T)$: In this case, $\mathrm{pr}_T(P)$ lies in the complement of the discs around the points $T_i$ with radius $\|\mathrm{bar}_T - T_i\|$, as illustrated in Figure 4.12(b). Since $P$ is projected to $\mathrm{pr}_T(P)$ outside $T$, there exists a point $B$ contained in an edge $(T_i, T_j)$ of $T$, such that

$$\|P - B\| \leq \|P - Q\|$$

(namely the intersection point of the line $(\mathrm{pr}_T(P), Q)$ with one of the triangle edges). Since $\|P - Q\| \leq \frac{s}{2}$, we can deduce that

$$\|P - B\| \leq \frac{s}{2}. \tag{4.6}$$

From (4.6) it also follows that $\|\mathrm{pr}_T(P) - B\| \leq \frac{s}{2}$, and moreover one of the norms $\|T_i - B\|$, $\|T_j - B\|$ is smaller than $\frac{s}{2}$. Therefore, we obtain

$$\begin{aligned} d(\mathrm{pr}_T(P), T) &\leq \sqrt{\min\{\|T_i - B\|^2, \|T_j - B\|^2\} + \|\mathrm{pr}_T(P) - B\|^2} \\ &\leq \frac{1}{2}\sqrt{2}s\ . \end{aligned} \tag{4.7}$$

Let $A$ be the image of $\mathrm{bar}_T$ under reflection along the edge $(T_i, T_j)$. Since $\mathrm{pr}_T(P)$ lies in the complement of the discs around the points $T_i$ with radius $\|\mathrm{bar}_T - T_i\|$, we have

$$\|B - \mathrm{pr}_T(P)\| \geq \left\|A - \frac{1}{2}(T_i + T_j)\right\| = \left\|\mathrm{bar}_T - \frac{1}{2}(T_i + T_j)\right\| = \frac{1}{6}\sqrt{3}s \tag{4.8}$$

Altogether, using Lemma 4.5.9, we obtain

$$
\begin{aligned}
d(P,T) &= \sqrt{\|P - \mathrm{pr}_T(P)\|^2 + d(\mathrm{pr}_T(P),T)^2} \\
&= \sqrt{\|P - B\|^2 - \|B - \mathrm{pr}_T(P)\|^2 + d(\mathrm{pr}_T(P),T)^2} \\
&\underbrace{\leq}_{(4.6)(4.8)(4.7)} \sqrt{(\tfrac{1}{2}s)^2 - (\tfrac{1}{6}\sqrt{3}s)^2 + (\tfrac{1}{2}\sqrt{2}s)^2} \\
&= \sqrt{\tfrac{2}{3}}s.
\end{aligned}
$$

Altogether, we have found a point $P$ of triangle $S$ which is closer to $T$ than $\sqrt{\frac{2}{3}}s$, and the assertion follows. □

# 5 Medial axes shape spaces and segmentation of 3D voxel data

Joint work with O. Aichholzer, S. Colutto, B. Kornberger and O. Scherzer.

Submitted to European Journal of Applied Mathematics.

This paper deals with supervised segmentation of topologically complex objects. Its focus is on modelling the prior shape space by medial axis representations, i.e. sets of medial center points and radii. Procrustes analysis is used to compute a mean shape and determine deviations between shapes. Skin surfaces are taken to define the implied boundary of a shape. The information is plugged into a functional of type (1.58) for segmentation.

As main contribution of this paper, topologically complex objects can be segmented using a small amount of computational data (compared to [78] f.e.) but not requiring much user interaction and modelling as in [58, 91].

External Contributions: B. Kornberger provided the Section on the medial axis transform. S. Colutto provided parts of the numerical implementation of the algorithm. O. Aichholzer and O. Scherzer participated in discussions and revised the paper.

# MEDIAL AXIS SHAPE SPACES AND SEGMENTATION OF 3D VOXEL DATA

Jochen Abhau [†], Oswin Aichholzer [‡],
Sebastian Colutto [♮],
Bernhard Kornberger [‡],
Otmar Scherzer [†,◇].

[†] Computational Science Center, University of Vienna
Nordbergstraße 15, A-1090 Wien, Austria
{jochen.abhau, otmar.scherzer}@univie.ac.at.

[‡] Institute for Software Technology,
Graz University of Technology, Austria
{oaich, bkorn}@ist.tugraz.at.

[♮] Institute of Mathematics,
Technikerstraße 21a, A-6020 Innsbruck, Austria
Sebastian.Colutto@uibk.ac.at.

[◇] Radon Institute of Computational and Applied Mathematics, Austrian Academy of Sciences,
Altenberger Straße 69, A-4040 Linz, Austria.

**Abstract.** In this paper we construct a shape space of medial ball representations from given shape training data using methods of Computational Geometry and Statistics. The ultimate goal is to employ the shape space as prior information in supervised segmentation algorithms for 3D voxel data. The construction of the shape space, the statistics of the training data, and their subsequent use in segmentation require up-to-date, respectively novel, methods to standardize the medial ball representation of the shape space. Aside from novel methods the whole pipeline of shape space, statistics, and segmentation is novel to the best of our knowledge.

**Key Words.** Medial ball representation, medial axis transform, image segmentation, Procrustes analysis, skin surfaces, EM algorithm.

## 1. INTRODUCTION

Segmentation and detection of objects in image data is of major importance in several applications, such as for instance medical imaging, video surveillance and motion tracking.

Segmentation methods based on minimization of energy functionals have proven to be very efficient. In this context it is common to differ between edge based [116] and region based energy functionals [87]. Another distinctive feature of segmentation algorithms is whether they perform supervised and unsupervised. Historically, unsupervised

segmentation algorithms have been introduced first. However, the segmentation results are poor if the data is noisy, has little contrast, or if the object is partially occluded. To cope with these difficult situations supervised segmentation is performed, which commonly is implemented by incorporating a shape statistics term in the energy functional for unsupervised segmentation.

In the following we review some energy minimization segmentation approaches which incorporate shape prior information:
A compact representation of shapes, called *M-reps* (Medial atom representation) has been proposed in [49] and [92]. An M-rep consists of medial atoms aligned on a regular grid structure. Medial atoms consist of a coordinate position in space, a radius, a local coordinate frame, and an angle. The atoms are samples of the medial axis [18]. The actual surface associated with the M-rep is an enfolding B-spline of the medial atoms. M-reps can be regarded as elements of a Riemannian shape manifold, which defines distances between M-reps by lengths of geodesics amongst them. An M-rep shape space is suitable for modeling shapes of very similar geometry and topology. Image segmentation with M-reps has been considered in [58, 91]. Mumford-Shah energy segmentation on an M-rep space has been presented in [32].

In [36] (this work is actually in 2D) contours are represented as simple, closed B-spline curves. The shape prior and statistic are computed as the mean and the principal components of the spline control points of the training shapes. For segmentation a Mumford-Shah like energy functional is supplemented by the Mahalanobis distance to the shape prior, which is calculated from the statistics. This approach is based on the assumption that the shapes are distributed according to a multivariate Gaussian distribution. Using kernel space techniques, a generalization to non-Gaussian distributed shapes is given in [35]. Applications to medical MR image segmentation have been studied in [28].

A level set approach for segmentation with shape priors has been studied in [78]. There shapes are associated with the associated signed distance functions. On the training distance functions of the shapes the actual statistics, which is used in the segmentation algorithm, is calculated. Shape statistics are implemented in a geodesic active contour evolution as maximum a posteriori estimator of the shape. By this approach geometrically and topologically complex shape priors can be considered. However, this representation requires large computational resources, and efficient implementations with narrow bands tend to create inaccuracies and artifacts in the segmentation results. In [44], the work of [78] has been improved concerning efficiency. The same shape prior strategy as in [78] has been used in combination with the unsupervised Chan-Vese energy minimization [27] in [113]. Further generalizations of [78] are given in [97] and [98].

In [30], shape statistical prior information is incorporated in a variational segmentation functional with an additional regularization term. A generalization of this work is presented in [51]. More recently, the paper [23] proposes a variational level set framework that takes into account shape prior information which combines gradient and region based segmentation.

In this paper, we consider the construction of *medial ball shape spaces* and the com-

putation of statistics of training shapes, which are then used for supervised energy minimization segmentation. We are given training shapes, which are represented as triangular surface meshes. Such can either be provided by expert segmentation, or have been obtained from unsupervised segmentation (see e.g. [3, 32]). For the latter one can also imagine to use data of other imaging modalities, which have higher contrast or are less affected by noise.

The following scheme, which can as well serve as an *outline* of our work, is studied:

**5.1.1 Scheme.**
**Preprocessing (Shape Statistics)**

- For each mesh a discrete medial axis transform

$$M = (x_1, \dots, x_k; r_1, \dots, r_k) \tag{5.1}$$

  is computed. We refer to $c(M) = (x_1, \dots, x_k) \in \mathbb{R}^{3\times k}$ as the *c*enters of the medial balls of $M$. It is important for our application that each discrete medial axis transform $M_j$ consists of the same number $k$ of medial balls $(x_i, r_i) \in \mathbb{R}^3 \times \mathbb{R}^+$. The algorithm developed in here is capable to do so, which is a novel aspect in the literature. For sake of simplicity, the discrete medial axis transform computed in this step will be called *ball representation* according to its definition (Section 5).

- A labeling of the ball representation $M_i$ is computed by an EM-algorithm and the Kuhn-Munkres algorithm (Section 5).

- Using Procrustes analysis, a mean ball representation is computed from the labeled ball representations, and a Mahalanobis distance between ball representations is defined. These are the building blocks of our medial ball shape space. (Section 5).

**Segmentation**

- For segmentation the shape space is implemented as shape prior information in a simplified Mumford-Shah functional. The implied surface of the ball representations is constructed by a skin surface [41] (Section 5).

The following case examples for segmentation are considered. In Section 5 the full pipeline of statistics and segmentation is applied to medical prostate data and synthetic data, and Section 5 concludes the paper.

# 2. MEDIAL AXIS TRANSFORM

In this section we consider the problem of approximating a number of bounded open sets $\Omega_1, \dots, \Omega_n \subset \mathbb{R}^3$ by sets $B_1, \dots, B_n$ of approximate medial balls, where all sets $B_i$ have the same cardinality. Stability of the approximation, described below, is of

vital importance for the statistical analysis of medial axis representations. Our novel approach for calculating sets of approximate medial balls of the same cardinality for a class of objects combines three well known methods. First we utilize Voronoi diagrams to approximate objects by sets of (approximate) medial balls [11, 12]. As these sets will usually have a rather huge cardinality we then use set covering methods to obtain sufficiently small and stable subsets [8, 7]. Finally, we use a $k$-means [59] clustering algorithm to control the cardinality of the approximating sets in order to obtain sets of approximate medial balls of uniform cardinality.

We start by reviewing some basic facts of the medial axis transform.

## 2.1. Discrete medial axis transform

The following definitions of the medial axis and medial axis transform are standard and can for instance be found in [19].

**5.2.2 Definition** ([19])**.** The inner *Medial Axis* $\mathrm{M}(\Omega)$ of a bounded open set $\Omega \in \mathbb{R}^3$ is the set of points $x \in \Omega$ which have at least two closest points in $\partial\Omega$, where $\partial\Omega$ denotes the boundary of $\Omega$. The inner *Medial Axis Transform* $\mathrm{MT}^{\mathrm{in}}(\Omega)$ is the collection of maximal (with respect to inclusion) open balls centered at $\mathrm{M}(\Omega)$ and included in $\Omega$.

Note that the cardinality of the inner Medial Axis Transform is in general infinite. Moreover the medial axis $\mathrm{M}(\Omega)$ behaves unstable with respect to perturbations of high curvature of $\partial\Omega$. That is, the medial axis might contain branches that are far from being intuitive. The corresponding inner medial axis transform $\mathrm{MT}^{\mathrm{in}}(\Omega)$ expresses this unstable behavior by small balls near the surface of $\Omega$. For these reasons we seek for an approximation of $\mathrm{MT}^{\mathrm{in}}(\Omega)$, which we define next.

**5.2.3 Definition** ([19])**.** The inner *Discrete Medial Axis Transform* $\mathrm{DMT}(\Omega)^{\mathrm{in}}$ is a finite set of open, approximate medial balls, which approximates the inner Medial Axis Transform $\mathrm{MT}^{\mathrm{in}}(\Omega)$.

In the following we assume that the input object $\Omega$ is given by a set $S_\Omega$ of sample points of the boundary $\partial\Omega$ of $\Omega$ and a triangular mesh $T(S_\Omega)$, representing $\partial\Omega$. We first show how to approximate **one** three-dimensional object $\Omega$ by the union of its discrete medial axis transform $\mathrm{DMT}(\Omega)^{\mathrm{in}}$ in a stable way. We start with the well known Voronoi approach [11, 12].

**5.2.4 Definition.** The *Voronoi Diagram* of a set of points $S_\Omega$ in $\mathbb{R}^3$ is a partition of $\mathbb{R}^3$ into (possibly unbounded) convex polyhedral regions, called Voronoi cells, such that each point $s_i \in S_\Omega$ has an associated Voronoi cell $v(s_i)$ with

$$v(s_i) := \{x \in \mathbb{R}^3 : \|x - s_i\| \leq \|x - s_j\|, \forall i \neq j,\ s_i, s_j \in S_\Omega\}.$$

We first compute the Voronoi diagram of $S_\Omega$. Then we extract for each sample point $s_i \in S_\Omega$ the inner pole point $p_i$, which is the vertex of the Voronoi cell $v(s_i)$ being farthest away from $s_i$ and inside $\Omega$. Finally, we construct for each inner pole point $p_i$ a

so called polar ball $B_{p_i,\rho_i}$ centered at $p_i$ with radius $\rho_i = \|s_i - p_i\|$. The set of all polar balls created in this way is the inner discrete medial axis transform $\mathrm{DMT}(\Omega)^{\mathrm{in}}$, and $S_\Omega$ is contained in the boundary of its union [11, 12]. See Figure 5.1(a) for an example with more than 10000 balls.

In the original approach a dense sampling $S_\Omega$ of a smooth surface $\partial\Omega$ of $\Omega$ is required in order to be able to distinguish inner Voronoi vertices from outer ones [11, 12]. To overcome this restriction we use, as mentioned above, the triangular surface mesh $T(S_\Omega)$ as additional input. This allows us to easily distinguish between inner and outer Voronoi vertices. Thus noise - always present in real-world data sets - and poor sampling quality do not affect the correctness of our inner/outer labeling, and correct operation of this step is ensured.

The centers of $\mathrm{DMT}(\Omega)^{\mathrm{in}}$ are close to $\mathrm{M}(\Omega)$, see [9] for a quantitative analysis and a precise statement of that fact. This implies that $\mathrm{DMT}(\Omega)^{\mathrm{in}}$ might include small, surface near balls corresponding to unwanted features of $\mathrm{MT}^{\mathrm{in}}(\Omega)$. Moreover, $\mathrm{DMT}(\Omega)^{\mathrm{in}}$ approximates $\Omega$ by up to $|S_\Omega|$ balls. For our purposes the instability and the high cardinality of $\mathrm{DMT}(\Omega)^{\mathrm{in}}$ prevent its direct usage.

To avoid these disadvantages we apply a pruning algorithm to $\mathrm{DMT}(\Omega)^{\mathrm{in}}$ in order to extract a proper subset of $\mathrm{DMT}(\Omega)^{\mathrm{in}}$. The result will be a stable (we remove surface near balls which result from instability) and compact representation of $\Omega$, cf. Figure 5.1(d) for an example. Moreover the pruning step will give us control to obtain sets of approximate medial balls of predefined cardinality, see Subsection 5 for details. In the following we briefly describe the pruning approach of [7], see there for further details.

By the above construction each ball in $\mathrm{DMT}(\Omega)^{\mathrm{in}}$ has four sample points on its boundary but none in its interior. We enlarge each ball of $\mathrm{DMT}(\Omega)^{\mathrm{in}}$ by a sufficiently small constant $\varepsilon > 0$. Then we use a spatial search structure to find for each enlarged ball $b_i$ all sample points from $S_\Omega$ which are now covered by $b_i$ (typically tens or even hundreds of sample points are contained). Finally we use a set-covering algorithm to find an (almost) minimal subset $\mathrm{DMT}(\Omega)^{\mathrm{in}}_*$ of the enlarged balls whose union covers all sample points $S_\Omega$. This set $\mathrm{DMT}(\Omega)^{\mathrm{in}}_*$ is the output of the pruning step.

Let us stress the fact that the applied set-covering algorithm to find a minimal subset of balls will favor balls that cover a large fraction of $S_\Omega$ and thus large areas of $\partial\Omega$ in order to make the cardinality of $\mathrm{DMT}(\Omega)^{\mathrm{in}}_*$, $\left|\mathrm{DMT}(\Omega)^{\mathrm{in}}_*\right|$, as small as possible. These balls are centered near stable parts of $\mathrm{M}(\Omega)$. This implies that surface-near balls, which originate from small perturbations of high curvature of $\partial\Omega$ (recall the above discussion) are avoided. Thus stability of our approach is greatly improved by the set-covering algorithm, see also [7]. In addition the one sided Hausdorff distance from the union of the obtained compact discrete medial axis transform $\mathrm{DMT}(\Omega)^{\mathrm{in}}_*$ to the original object $\Omega$ is bounded by $O(\varepsilon)$.

## 2.2. Sets of approximate medial balls of uniform cardinality

As described in Section 5 our goal is to obtain sets of approximate medial balls of uniform cardinality for different input objects, $\Omega_i$, $i = 1, \ldots, n$. So far we have shown

how to compute a stable representation $\mathrm{DMT}(\Omega_\mathrm{i})^{\mathrm{in}}_*$ of the enlarged discrete medial axis transform for one fixed object $\Omega_i$. We can control $|\mathrm{DMT}(\Omega_\mathrm{i})^{\mathrm{in}}_*|$ to some extent by the value $\varepsilon$ by which we enlarge the polar balls.

For each single object $\Omega_i$, the value $\varepsilon$ is chosen such that the resulting set $|\mathrm{DMT}(\Omega_\mathrm{i})^{\mathrm{in}}_*|$ is at least as large as the desired uniform cardinality $k$, that is, $k \leq \min_i(|\mathrm{DMT}(\Omega_\mathrm{i})^{\mathrm{in}}_*|)$. This approach is justified by the fact, that for piecewise linear $\partial\Omega$, the quantity $|\mathrm{DMT}(\Omega_\mathrm{i})^{\mathrm{in}}_*|$ tends to infinity if $\varepsilon \to 0$. We then apply the $k$-means clustering algorithm to each $\mathrm{DMT}(\Omega_\mathrm{i})^{\mathrm{in}}_*$ and get for every set $\mathrm{DMT}(\Omega_\mathrm{i})^{\mathrm{in}}_*$ exactly $k$ clusters. For each cluster we choose as an representative ball the one whose center is closest to the center of that cluster. This results in $n$ stable representations of $\{\Omega_1, \ldots, \Omega_n\}$ by sets of balls $\{B_1, \ldots, B_n\}$ with uniform cardinality $k$.

For the remainder of the paper we will use the term *ball representation* to refer to the pruned inner discrete medial axis transform of uniform cardinality.

## 3. LABELING OF THE BALL REPRESENTATIONS

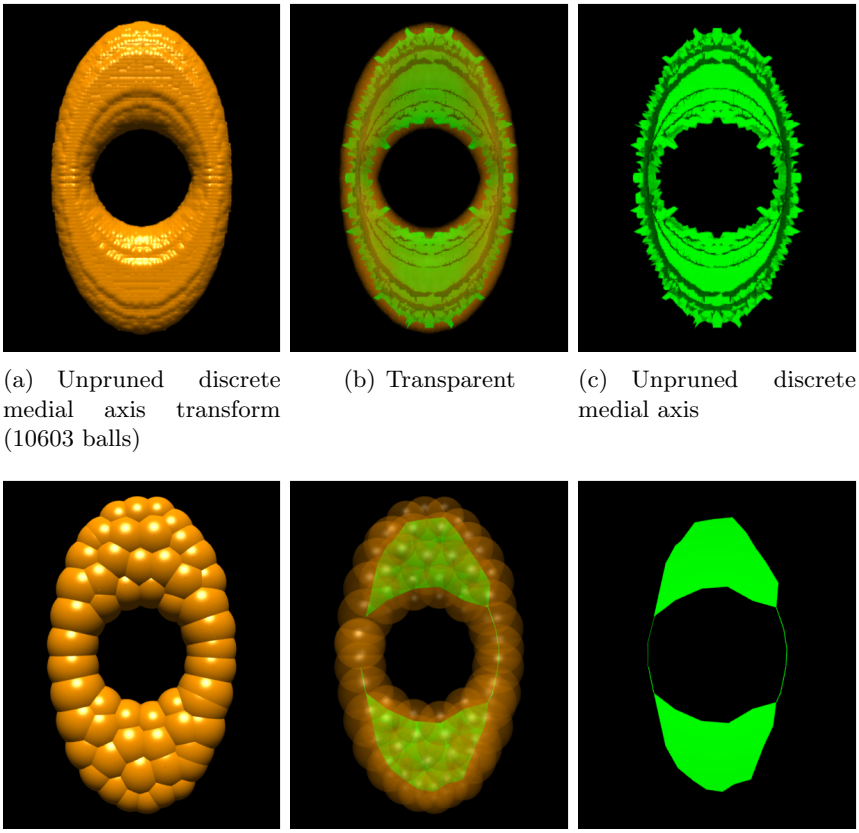
In this section we use statistical algorithms for labeling of the ball representations. Given two ball representations $M = (x_1, \ldots, x_k; r_1, \ldots, r_k)$ and
$\tilde{M} = (\tilde{x}_1, \ldots, \tilde{x}_k; \tilde{r}_1, \ldots, \tilde{r}_k)$, the labeling problem consists in determining a rearrangement of indices such that the two ball representations optimally match. Every matching involves a distance measure between ball representations. In view of the used definition of an Euclidean invariant shape space of ball representations considered below (see Section 5), we consider the following minimization problem for determining the optimal alinement:

$$\min_{\rho>0, A\in SO(3), b\in\mathbb{R}^3, \pi\in\mathcal{S}^k} \sum_{i=1}^{k} \|\rho A x_{\pi(i)} - b - \tilde{x}_i\|^2 + \alpha(\rho r_{\pi(i)} - \tilde{r}_i)^2 \,. \tag{5.2}$$

Here $SO(3)$ is the special orthogonal group of degree 3 (that is the group of rotations in $\mathbb{R}^3$, which can be represented as unitary matrices with determinant one), $\mathcal{S}^k$ is the symmetric group of degree $k$, and $\|\cdot\|$ denotes the Euclidean norm in $\mathbb{R}^3$. The parameter $\alpha$ is a weighting parameter, providing a trade-off between matching of balls and radii. Since it is difficult to compute global minimizers of (5.2) efficiently, we use a two-step algorithm from [64], to compute approximate minimizers. This algorithm consists of the following two steps:

- An Expectation Maximization (EM) algorithm is used to compute a similarity transformation, i.elet@tokeneonedota translation $b \in \mathbb{R}^3$, a rotation $A \in SO(3)$, and a scaling $\rho > 0$, which is close to an optimal one in (5.2). The algorithm is explained in Subsection 5.

- Keeping the similarity transform fixed, the Kuhn-Munkres algorithm is used to



(a) Unpruned discrete medial axis transform (10603 balls)

(b) Transparent

(c) Unpruned discrete medial axis

(d) Pruned discrete medial axis transform (55 balls)

(e) Transparent

(f) Pruned discrete medial axis

Figure 5.1: Discrete Medial Axis Transforms and Discrete Medial Axes

compute an optimal labeling $\pi \in \mathcal{S}^k$ in (5.2). This algorithm is explained in Subsection 5.

## 3.1. EM-algorithm to compute the similarity transformation

EM-algorithms have been used successfully for solving a wide range of labeling problems [63]. In general, EM algorithms [85] consist in maximizing a likelihood function, which depends on parameters and hidden variables. There two successive steps are performed iteratively:

- An **E**xpectation step, which consists in computation of the expected values of the hidden variables and
- a **M**aximization step, which is to optimize the likelihood function for the parameters.

We propose an algorithm which is closely related to [64]. However, the difference is that we take into account information on the radii of the medial balls in addition. The key idea of [64] is to compute the optimal similarity transformation of (5.2) relaxing the constraint $\pi \in \mathcal{S}^k$. This allows to apply the EM algorithm.

1. *Relaxation step.*

   A mapping $\pi \in \mathcal{S}^n$ can be represented as a matrix $P \in \{0,1\}^{k\times k}$ with entries

   $$P_{j,i} = \begin{cases} 1 & \text{if } \pi(j) = i \\ 0 & \text{otherwise} \end{cases} .$$

   The relaxation consists in assuming instead of $\mathcal{S}^k$ *stochastic matrices*

   $$\{P \in [0,1]^{k\times k} : \sum_{i=1}^{k} P_{j,i} = 1 \text{ for all } j\} .$$

   It is common to interpret $P_{j,i}$ as the probability that $j$ is mapped to $i$. For medial ball representation this means that the ball $(x_j, r_j)$ is mapped onto $(\tilde{x}_i, \tilde{r}_i)$.

   Furthermore, a distribution $p = (p_1, \ldots, p_k)$ is assumed over $\{1, \ldots, k\}$. The number $p_i$ represents the probability, that an arbitrary element of $\{1, \ldots, k\}$ is mapped to $i$.

2. *Distances of balls.*

   Under the condition $\pi(j) = i$ and for a given similarity transformation $(\rho, A, b)$, we assume that the transformed ball $(x_j, r_j)$ is normally distributed around $(\tilde{x}_i, \tilde{r}_i)$, i.e. $\rho A x_j + b$ is normally distributed with mean $\tilde{x}_i$ and $\rho r_j$ is normally distributed with mean $\tilde{r}_i$. Concerning the radii, this assumption is slightly inaccurate since radii cannot become negative; we overcome this problem by choosing the variance

$\sigma$ sufficiently small, such that the 0.01 quantile is positive. Hence we define (for $i$, $j = 1, \ldots, k$)

$$g_i(x_j, r_j) = (2\pi\sigma^2)^{-\frac{1}{2}} \exp\left(-\frac{1}{2\sigma^2}\left(||\rho A x_j + b - \tilde{x}_i||_2^2 + \alpha(\rho r_j - \tilde{r}_i)^2\right)\right) . \quad (5.3)$$

The function $g_i(x_j, r_j)$ is large if $(x_j, r_j)$ is mapped closely to $(\tilde{x}_i, \tilde{r}_i)$ and small otherwise.

The likelihood function for the parameters of the similarity transform $(\rho, A, b)$ and the relaxed labeling $(P, p)$ for given ball representations $M$ and $\tilde{M}$ is then given by the product probability

$$L(\rho, A, b, P, p) = \prod_{i,j=1}^{k} (p_i g_i(x_j, r_j))^{P_{j,i}} . \quad (5.4)$$

Taking the logarithm on both sides defines the log-likelihood function

$$l(\rho, A, b, P, p) = \sum_{i,j=1}^{k} P_{j,i} \left(\log p_i + \log g_i(x_j, r_j)\right) . \quad (5.5)$$

It is important to note, that if $P$ is a permutation, then $p_i = 1/k$. Therefore in this case, maximizers of (5.5) are minimizers in (5.2) and vice versa. Following [64], enlarging the class of labelings from $\mathcal{S}^k$ to stochastic matrices has little effect on the optimal similarity transform.
The advantage of the formulation (5.5), compared to (5.2), is, that (5.5) can be minimized efficiently with an EM-algorithm, where the variables $\rho, A, b, p$ are regarded as parameters, and the entries of $P$ are regarded as hidden variables of the likelihood function $l$. The EM algorithm for minimization of (5.5) reads as follows:

**Algorithm 1** EM Maximization of $l$

Initialize $\rho = 1$, $A = id$ and $b = 0 \in \mathbb{R}^3$. Choose values for $\sigma$, $\alpha$.
**while** $l$ still increases **do**
**E-Step**: Compute the expected value of the relaxed permutation matrix $P$,

$$P_{j,i} \leftarrow \frac{p_i g_i(x_j, r_j)}{\sum_{s=1}^{k} p_s g_s(x_j, r_j)} \quad (5.6)$$

By this assignment it is guaranteed that $\sum_{i=1}^{k} P_{j,i} = 1$ for all $j$.

**M-Step**: Maximize

$$\sum_{i,j=1}^{k} P_{j,i}(\log p_i + \log g_i(x_j, r_j)) \quad (5.7)$$

over $A$, $\rho$, $b$ and $p_1, \ldots, p_k$.
Compute the new value of $l$.
**end while**

The formulas to compute the maximum in (5.7) and their derivation are given in the appendix.

Compared to different algorithms for minimizing (5.2) such as Markov Chain Monte Carlo Methods (MCMC), the output of the EM Algorithm is more sensitive to initial values of the parameters. This is no drawback in our case, since the ball representations are often already quite well aligned. However, the provided algorithm is assumed to be more efficient than MCMC.

### 3.2. Labeling

This subsection is concerned with computing an optimal labeling $\pi \in \mathcal{S}^k$ in (5.2).

It is instructive to reformulate the labeling problem as a combinatorial optimization problem.

1. A weighted graph is constructed of the following data:
   - Nodes are the $2k$ balls $(x_1, r_1) \ldots, (x_k, r_k)$ and $(\tilde{x}_1, \tilde{r}_1), \ldots, (\tilde{x}_k, \tilde{r}_k)$.
   - Let $\alpha$ be as in (5.2). The weights of the edges between the nodes are defined by
     $$w_{j,i} = ||\rho A x_j - b - \tilde{x}_i)||^2 + \alpha(\rho r_j - \tilde{r}_i)^2, \qquad \text{with } \alpha \in [0, 1].$$
     Here, $(\rho, A, b)$ is the similarity transform computed in Subsection 5.
2. A permutation $\pi \in \mathcal{S}^k$ minimizing
   $$\sum_{i=1}^{k} w_{\pi(i),i} \tag{5.8}$$
   is computed.

For the fixed similarity transform transformation $(\rho, A, b)$, a minimizer $\pi \in \mathcal{S}^k$ of (5.2) is a minimizer of (5.8) and vice versa. For the solution of the matching problem (5.8) we apply the Kuhn-Munkres algorithm, which is a special case of the primal-dual algorithm in linear programming [6]. According to [107] the complexity of the algorithm is $O(k^3)$. In our applications, the high order of complexity of the algorithm is not an issue since it is only applied for medial ball representations of dimension $k$, which is in the range of 10 to 50.

## 4. MEDIAL BALL SHAPE SPACE BY PROCRUSTES ANALYSIS

In the following we perform a statistics of the ball representations $M_1, \ldots, M_n$ and establish a shape space, which we call *medial ball* shape space. Moreover, a distance between elements of the medial ball shape space is employed to compare them quantitatively.

## 4.1. Elements of the medial ball shape space

A shape is all the geometrical information that remains when location, scale and rotational effects are filtered out from an object (taken from [40, Chapter 1]). We consider this definition when constructing the medial ball shape space as the factor space of ball representations $M = (x_1, \ldots, x_k; r_1, \ldots, r_k)$ modulo similarity transformations. That is

$$\Sigma = \mathbb{R}^{3\times k} \times (\mathbb{R}^{>0})^k / \sim_{ST} \ . \tag{5.9}$$

That is, two ball representations $M = (x_1, \ldots, x_k; r_1, \ldots, r_k)$ and $\tilde{M} = (\tilde{x}_1, \ldots, \tilde{x}_k; \tilde{r}_1, \ldots, \tilde{r}_k)$ represent the same shape (or in other words are equivalent with respect to $\sim_{ST}$) if there exists a similarity transformation $(\rho, A, b)$ such that

$$\rho A\tilde{x}_i + b = x_i \text{ and } \rho\tilde{r}_i = r_i \text{ for } i = 1, \ldots, k \tag{5.10}$$

We emphasize that in the segmentation literature (see e.g. [32, 91, 78, 36]) frequently shape spaces are not defined as factor spaces. As a consequence the according shape space allows for different representations of the same shape.

Below we derive a segmentation model taking into account the original shape concept and coordinates of Kendall [62]. Compared to Bookstein [20], Kendall coordinates do not depend on the choice of a baseline i.e. two medial balls in our case. Therefore statistical analysis like PCA is not distorted. Since it is bulky to define a distance between equivalence classes we systematically choose representatives, for which we define a shape distance. The process is similar to [40, Chapter 4] where mid axis representations $M = (x_1, \ldots, x_k)$ (without radii) have been considered.

1. A *mean* ball representation $\boldsymbol{\mu}$ of the (labeled) ball representations $M_1, \ldots, M_n$ is defined by

$$\boldsymbol{\mu} = \underset{\tilde{\boldsymbol{\mu}}=\{(\tilde{\mu}_1,\tilde{r}_1),\ldots,(\tilde{\mu}_k,\tilde{r}_k)\}}{\arg\min} \sum_{l=1}^{n} d(M_l, \tilde{\boldsymbol{\mu}})^2 \tag{5.11}$$

   where

$$d(M_l, \tilde{\boldsymbol{\mu}})^2 = \min_{\rho>0, A\in SO(3), b\in\mathbb{R}^3} \sum_{i=1}^{k} \|\rho A x_i^{(l)} - b - \tilde{\mu}_i\|^2 + \alpha(\rho r_i^{(l)} - \tilde{r}_i)^2 \tag{5.12}$$

   The parameter $\alpha$ is the same as in Equation (5.2). The minimization problem (5.11) can be solved with an iterative algorithm, which is along the lines of [40, Chapter 5].

2. $M$ is normalized with respect to *translation* by multiplying the ball centers of $M$ with the $(k-1)\times k$ *Helmert submatrix* $H$ (see [40, p.34] for the definition). The advantage in using the Helmert matrix for this task (compared to for instance centering the coordinates) is that the arising representation only consists of $k-1$ centers and that the Helmert matrices can be computed efficiently inductively. The resulting representation $T(M)$ is given by

$$T(M) = ((H[c(M)^T])^T; r_1, \ldots, r_k) \in \mathbb{R}^{3\times(k-1)} \backslash \{0\} \times (\mathbb{R}^{>0})^k \ . \tag{5.13}$$

3. $T(M)$ is normalized with respect to *scaling* by dividing the coordinates and radii by the Frobenius norm $\|c(T(M))\|$. The resulting scaled representation is denoted by $S(T(M))$.

4. $S(T(M))$ is normalized with respect to *Rotation* by minimizing the Frobenius norm

$$\min_{A \in SO(3)} \|c(S(T(\boldsymbol{\mu})))^T - c(S(T(M)))^T A\| \,. \tag{5.14}$$

Rotating $S(T(M))$ by the optimal $A \in SO(3)$ gives

$$R_{\boldsymbol{\mu}}(S(T(M))) = \left( c(S(T(M)))A; \frac{r_1}{\|c(T(M))\|}, \ldots, \frac{r_k}{\|c(T(M))\|} \right) , \tag{5.15}$$

which is the representative of the class.

As shown in [62], the optimal rotation $A$ in (5.14) can be calculated as follows: If

$$(c(S(T(\boldsymbol{\mu}))))c(S(T(M)))^T = V \Lambda U^T \tag{5.16}$$

is the signed singular value decomposition (see Appendix) of $(c(S(T(\boldsymbol{\mu}))))c(S(T(M)))^T$, then

$$A = UV^T. \tag{5.17}$$

The mean shape $\mu$ and the representative are utilized to compute distances in $\Sigma$ in the next Subsection 5.

## 4.2. Variability in the medial ball shape space

Standard techniques for capturing the variability of data are principal component analysis (PCA) and the Mahalanobis distance. Historically, these techniques have been defined in Euclidean space. A standard way to transfer these concepts to Riemannian manifolds is to project the data to a tangent space (which is Euclidean) and perform an variability analysis there. Recently, techniques have been established, which consider the whole tangent bundle for parallel transport of tangent vectors along geodesics and perform a *principal geodesic analysis*, [46]. In our case, the medial ball shape space $\Sigma$ is a Riemannian manifold with singularities. As is detailed in [103] for shape spaces of point configurations and carries over to our situation, $\Sigma$ has singularities at equivalence classes of ball representations under translation and scaling, where the rotation group does not act free. These shapes arise from ball representations consisting of balls of the same radius lying on a straight line. We can exclude these ball representations from our further considerations since they are not of interest to us, or apply the simulation of simplicity concept [42] to ball representations $M_1, \ldots, M_n$ and avoid this case. For sake of simplicity, we present only details on the projection technique, although the whole shape analysis and segmentation pipeline is generalizable to a nonlinear framework.

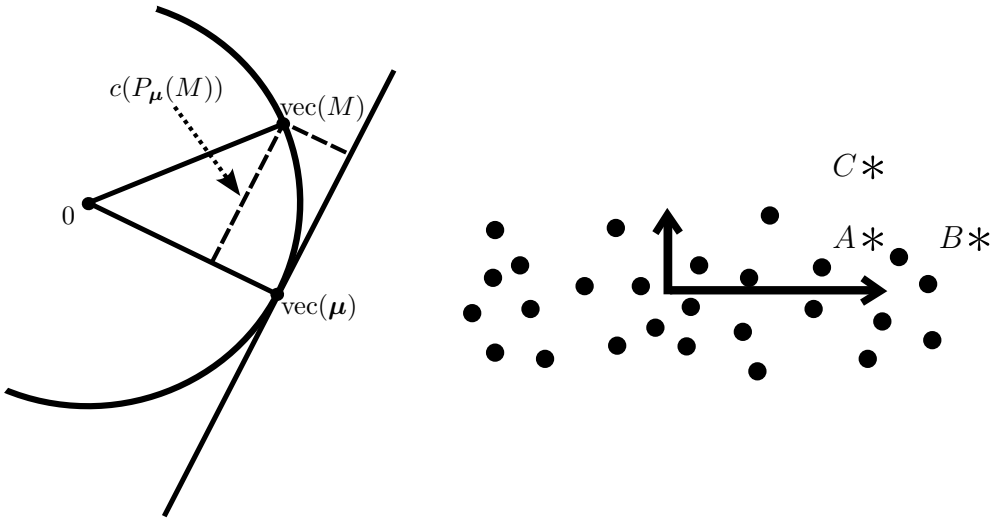



Figure 5.2: On the left hand side, Procrustes tangent coordinates of the centers of a ball representation are shown. Translated, scaled and rotated ball representations $\boldsymbol{\mu}$, $M$ are on the circle. vec$(M)$ is orthogonally projected to the subspace spanned by vec$(\boldsymbol{\mu})$. The difference between the projection of vec$(M)$ and vec$(\boldsymbol{\mu})$ is $c(P_{\boldsymbol{\mu}}(M))$. PCA and Mahalanobis distance are visualized on the right hand side. The data points vary in horizontal direction much more than in vertical direction. The Mahalanobis distance between $A$ and $B$ is smaller than the distance between $A$ and $C$, which is the opposite measured in the Euclidean distance.

*The tangent space of* $\Sigma$ *at* $[\boldsymbol{\mu}]_{\sim_{ST}}$.
For our medial ball shape space $\Sigma$, we compute projections of shapes on the tangent space at $[\boldsymbol{\mu}]_{\sim_{ST}}$. Let $M = (x_1, \ldots, x_{k-1}; r_1, \ldots, r_k)$ be a ball representation, in normalized form (5.15), and vec$(M) = (x_1^T, \ldots, x_{k-1}^T)^T \in \mathbb{R}^{3k-3}$. Then the orthogonal projection of $M$ to the tangent space of $\Sigma$ at $\boldsymbol{\mu}$ is given by

$$P_{\boldsymbol{\mu}}(M) = (\text{vec}(M) - \langle \text{vec}(M), \text{vec}(\boldsymbol{\mu}) \rangle \text{ vec}(\boldsymbol{\mu}); r_1, \ldots, r_k) \in \mathbb{R}^{3k-3} \times (\mathbb{R}^{>0})^k \qquad (5.18)$$

Concerning the center points $c(M)$ of $M$, these are the Procrustes tangent coordinates as in [40, p.76]. The tangent projection is illustrated in Figure 5.2, left part. Tangent coordinates represent the linear deviation of $M$ relative to $\boldsymbol{\mu}$ and will be used for statistical analysis in the sequel.

*PCA and Mahalanobis distance for ball representations in* $\Sigma$.
First we recall the concepts of *principal component analysis (PCA)* and *Mahalanobis distance* in $\mathbb{R}^d$. For data points $m_1, \ldots, m_n \in \mathbb{R}^d$, a PCA is given by the normed

eigenvectors of the covariance matrix

$$\mathrm{Cov}(m_1,\ldots,m_n)=\frac{1}{n}\sum_{i=1}^{n}(m_i-\overline{m})(m_i-\overline{m})^T \qquad \text{with} \qquad \overline{m}=\frac{1}{n}\sum_{i=1}^{n}m_i. \tag{5.19}$$

The eigenvectors $e_1,\ldots,e_d$ are sorted by the size of the corresponding eigenvalues $\lambda_1,\ldots,\lambda_d$ and give an orthonormal basis of $\mathbb{R}^d$. This basis represents the main directions of variability of the data vectors $m_1,\ldots,m_n$.

A distance which takes into account the variability of the data $m_1,\ldots,m_n$ is given by the Mahalanobis distance. This distance is defined by

$$d_{\mathbb{R}^d}(\tilde{m}_1,\tilde{m}_2)=(\tilde{m}_1-\tilde{m}_2)^T(\mathrm{Cov}(m_1,\ldots,m_n))^{-1}(\tilde{m}_1-\tilde{m}_2) \tag{5.20}$$

for $\tilde{m}_1$ and $\tilde{m}_2\in\mathbb{R}^d$. Compared to the Euclidean distance, the Mahalanobis distance of two data points is small, if they differ along the main directions of variability (which are the principal components of $m_1,\ldots,m_k$), and it is large, if they differ along other directions. If $d$ is large it might be appropriate to use a PCA first, and taking only the $\tilde{d}$ most significant components $e_1,\ldots,e_{\tilde{d}}$ to eigenvalues $\lambda_1,\ldots,\lambda_{\tilde{d}}$. This results in a simplified Mahalanobis distance

$$\tilde{d}_{\mathbb{R}^d}(\tilde{m}_1,\tilde{m}_2)=(\tilde{m}_1-\tilde{m}_2)^T\tilde{U}\tilde{D}^{-1}\tilde{U}^T(\tilde{m}_1-\tilde{m}_2) \tag{5.21}$$

with $\tilde{U}=(e_1,\ldots,e_{\tilde{d}})$ and $\tilde{D}$ the diagonal matrix with entries $\lambda_1,\ldots,\lambda_{\tilde{d}}$.

Applying these concepts to ball representations, let $P_{\boldsymbol{\mu}}(M_i)=(c^i_{\boldsymbol{\mu}};r^i_{\boldsymbol{\mu}})\in\mathbb{R}^{3k-3}\times(\mathbb{R}^{>0})^k$ be the tangent coordinates of $M_i$, split into centers $c^i_{\boldsymbol{\mu}}$ and radii $r^i_{\boldsymbol{\mu}}$. A PCA of ball representations $M_1,\ldots,M_n$, is given by the eigenvectors and eigenvalues of the covariance matrices

$$\mathrm{Cov}(c^1_{\boldsymbol{\mu}},\ldots,c^n_{\boldsymbol{\mu}}) \text{ and } \mathrm{Cov}(r^1_{\boldsymbol{\mu}},\ldots,r^n_{\boldsymbol{\mu}}). \tag{5.22}$$

The Mahalanobis distance $d_\Sigma$ between $M$ and $\tilde{M}$ in $\Sigma$ is then given by
(with $P_{\boldsymbol{\mu}}(M)=(c_{\boldsymbol{\mu}};r_{\boldsymbol{\mu}})$ and $P_{\boldsymbol{\mu}}(\tilde{M})=(\tilde{c}_{\boldsymbol{\mu}};\tilde{r}_{\boldsymbol{\mu}})$)

$$\begin{aligned} d_\Sigma(M,\tilde{M}) &= (c_{\boldsymbol{\mu}}-\tilde{c}_{\boldsymbol{\mu}})^T\,\mathrm{Cov}\left(c^1_{\boldsymbol{\mu}},\ldots,c^n_{\boldsymbol{\mu}}\right)^{-1}(c_{\boldsymbol{\mu}}-\tilde{c}_{\boldsymbol{\mu}}) \\ &\quad +\alpha\,(r_{\boldsymbol{\mu}}-\tilde{r}_{\boldsymbol{\mu}})^T\,\mathrm{Cov}\left(r^1_{\boldsymbol{\mu}},\ldots,r^n_{\boldsymbol{\mu}}\right)^{-1}(r_{\boldsymbol{\mu}}-\tilde{r}_{\boldsymbol{\mu}})\ . \end{aligned} \tag{5.23}$$

In case that the ball representations contain many balls, a PCA is performed first and a Mahalanobis distance as in (5.21) is computed.
PCA and Mahalanobis distance are illustrated in Figure 5.2, right part.

As a result of this section, a mean shape and distances between shapes have been defined and this statistics can be used as shape prior in segmentation in the sequel.

# 5. SEGMENTATION

In this section we consider object segmentation in 3D voxel data. Thereby we aim for taking into account shape prior information given in the form that the object to be recovered belongs to medial ball shape space. The distance on the shape space is given by the Mahalanobis distance (see Sections 5-5). The medial ball shape space is computed from boundaries of training objects as discussed in Section 5.

The boundary of a ball representation is defined as the *skin surface* [41]. Below we give a definition and some properties of skin surfaces which motivate this choice.

Finally we describe the whole pipeline of the proposed segmentation process.

## 5.1. Skin Surfaces

In this subsection, we review the definition and basic properties of skin surfaces. Moreover, we discuss properties of skin surfaces, which predestine them as boundary surfaces of medial ball representations.

**Definition of a skin surface.**
Basic building blocks of skin surfaces are *weighted points* $p = (x_p, w_p) \in \mathbb{R}^3 \times \mathbb{R}$. Weighted points can for instance represent balls with center $x$ and radius $r$, setting $x_p = x$ and $w_p = r^2$, and therefore a ball representation can be regarded as a set of weighted points. Addition and scalar multiplication of weighted points $p$, $q$ are defined by

$$p + q = (x_p + x_q, w_p + w_q + 2\langle x_p, x_q \rangle), \tag{5.24}$$

and

$$sp = (sx_p, sw_p + (s^2 - s)||x_p||^2) \quad \text{for} \quad s > 0. \tag{5.25}$$

Here, $\langle \cdot, \cdot \rangle$ and $|| \cdot ||$ denote the Euclidean scalar product and its induced norm. As usual, but with these algebraic operations, the convex hull of a set of weighted points $P = \{p_1, \ldots, p_k\}$ is defined by $\text{conv}(P) = \left\{ \sum_{i=1}^{k} \lambda_i p_i \mid \lambda_i \geq 0, \sum_{i=1}^{k} \lambda_i = 1 \right\}$. The *shrinkage* $p^s$ of a weighted point $p = (x_p, w_p)$ by a factor $s \geq 0$ is defined by $p^s = (x_p, sw_p)$.

Assume now we are given a set $P$ of weighted points $p = (x, w)$ with $w > 0$. Shrinkage applied to $P$ by $s \geq 0$ is defined by $P^s = \{p^s \mid p \in P\}$. Let $B_w(x) = \{y \in \mathbb{R}^3 \mid \|x - y\| \leq w\}$ be a ball with radius $w$ and center $x$. We denote by $\partial$ denote the boundary of a set in Euclidean space.

Let $s \geq 0$. Then the *s-skin* of $P$ is defined by

$$\text{skn}^s(P) = \partial \left( \bigcup_{q=(x_q, w_q) \in (\text{conv}P)^s} B_{\sqrt{w_q}}(x_q) \right) \tag{5.26}$$

Note that the square root of a negative number is imaginary and a ball with an imaginary radius is the empty set.

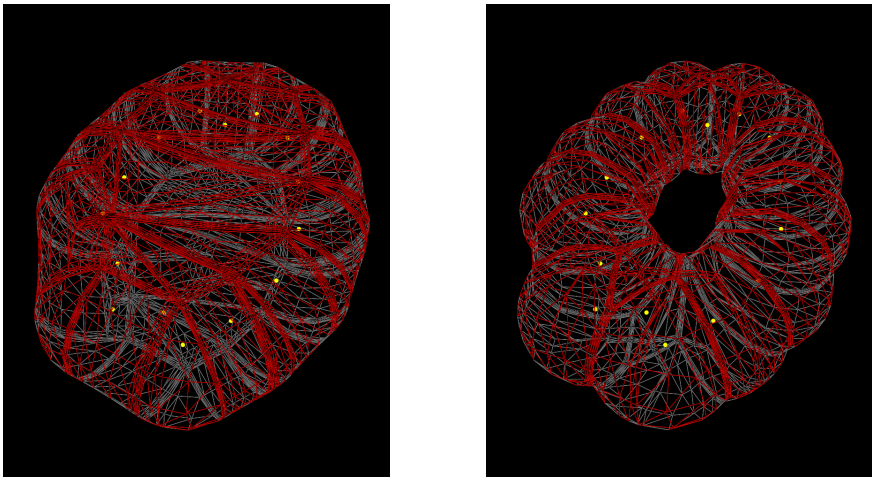


Figure 5.3: Examples of a skin surface of 15 balls with shrink factor $s = 0.1$ (left) and $s = 0.9$ (right). The dots indicate the centers of the medial balls. Note the topology changing effect of increasing the shrink factor, which can be utilized for constructing shapes of different topology.

In the special case $s = 0$, the skin surface of $p_1, \ldots, p_k$ reduces to the convex hull of $x_{p_1}, \ldots, x_{p_k}$, and in case $s = 1$, the skin surface is the boundary of the union of the input weighted points.

Since weighted points are shrunk by a factor of $s \in [0, 1]$ in the construction of the skin surface, the skin surface of a surface represented by medial balls is smaller than the original surface. A way to circumvent this problem is by prescaling the input weights $w_{p_1}, \ldots, w_{p_k}$ by $1/s$.

**Properties of skin surfaces.**
We employ the concept of a skin surface later on for definition of a surface from a ball representation. We believe that skin surfaces are very suitable for this task, since they have several attractive properties which we discuss in the sequel.

- *Information efficiency:* A skin surface only requires center points, weights and a shrinking factor for its complete description. Center points and weights are exactly the information available to us after (mean) ball representation computation in sections (5) and (5). The shrinking factor can be used as a tuning parameter depending on the application. Other popular surface construction methods like NURBS (or other splines), used for instance in [36], or Gregory patches (see [94, 32]) suffer from the drawback, that an additional grid structure of the control points is required for surface generation, which is difficult to obtain in an automatic way.
- *Computational efficiency:* There exists an efficient algorithm for computing skin

surfaces (see [68]). For $k$ weighted input points, the algorithm produces a triangular mesh consisting of $\mathcal{O}(k^2)$ vertices.

- *Ball representation fidelity:* If $M$ is a ball representation of some surface in $\mathbb{R}^3$, and $\mathrm{skn}(M)$ is the skin surface defined by $M$, then $M$ is contained in the ball representation of $\mathrm{skn}(M)$. This follows easily from the definition of the skin surface as a convex hull.
- *Economy:* A small number of input points can generate complicated skin surfaces, [41].
- *Universality:* Every orientable closed surface has a skin representation, [41].

An example of a skin surface, meshed with algorithm [68] is given in Figure 5.3.

**The implied skin surface of a shape.**
For a fixed ball representation $\boldsymbol{\mu}$, and some given similarity transformation $(\rho, A, b)$, a shape in $\Sigma$ implies a surface. For its computation, it is necessary to reverse the transformations in Subsection 5. To give some details, assume that the shape in $\Sigma$ is given by its tangent coordinates $P_{\boldsymbol{\mu}} = (c_{\boldsymbol{\mu}}, r_{\boldsymbol{\mu}})$. First note that the tangent projection (5.18) is injective, with inverse

$$(c_{\boldsymbol{\mu}}; r_{\boldsymbol{\mu}}) \mapsto \left(\mathrm{vec}^{-1}\left(\sqrt{1 - c_{\boldsymbol{\mu}}^T c_{\boldsymbol{\mu}}}\ \mathrm{vec}(c_{\boldsymbol{\mu}}(\boldsymbol{\mu})) + c_{\boldsymbol{\mu}}\right); r_{\boldsymbol{\mu}}\right) \tag{5.27}$$

mapping tangent coordinates to normalized ball representations. Normalized ball representations are then aligned in $\mathbb{R}^3$ by applying $A^{-1}$ and scaling with $1/\rho$. When reversing translation, note that the Helmert matrix $H$ has a right inverse $\tilde{H} \in \mathbb{R}^{(k-1)\times(k-1)}$.

Applying $\tilde{H}$ and translating the result by $b$, an (aligned) ball representation $M$ consisting of $k$ balls in $\mathbb{R}^3$ is obtained, to which the skin surface

$$\gamma(M) = \gamma(P_{\boldsymbol{\mu}}, \rho, A, b) \tag{5.28}$$

is the implied boundary.

## 5.2. Region and edge based segmentation

For segmentation of voxel images, there are essentially two types of segmentation methods. *Region based* segmentation is applied to images, if the mean intensity of voxels inside the object to segment differs significantly from the mean intensity outside the object. If contrasts are low and objects are only separated by curves, *gradient based* segmentation is used. Here, we briefly discuss both approaches. Let $\Omega \subset \mathbb{R}^3$ be a bounded domain, and $u : \Omega \to \mathbb{R}$ an image intensity function.

*Region based segmentation.*

For a closed surface $\gamma \subset \Omega$, let $\mathcal{I}(\gamma) \subset \Omega$ be the inner part of $\gamma$ and $\mathcal{O}(\gamma) \subset \Omega$ the outer

part of $\gamma$. The mean values of an image intensity function $u$ on $\mathcal{I}(\gamma)$ and $\mathcal{O}(\gamma)$ are then given by

$$\overline{u}_{\mathcal{I}}(\gamma) = \frac{1}{|\mathcal{I}(\gamma)|}\int_{\mathcal{I}(\gamma)} u\ dx \qquad \text{and} \qquad \overline{u}_{\mathcal{O}}(\gamma) = \frac{1}{|\mathcal{O}(\gamma)|}\int_{\mathcal{O}(\gamma)} u\ dx \tag{5.29}$$

A simplified version of the Mumford-Shah model introduced in [27] consists in minimization of the functional

$$I_{SMS}(\gamma) = \int_{\mathcal{I}(\gamma)} (\overline{u}_{\mathcal{I}}(\gamma) - u)^2\ dx + \int_{\mathcal{O}(\gamma)} (\overline{u}_{\mathcal{O}}(\gamma) - u)^2\ dx\ . \tag{5.30}$$

If we assume that the mean intensity of an object in image $u$ differs significantly from the mean intensity of the region outside of this object, the functional $I_{SMS}$ is minimized by a submanifold $\gamma$ which represents the boundary of the object to segment, and $\mathcal{I}(\gamma)$ is the object itself.

*Edge based segmentation.*

In edge based segmentation, an object and its closed boundary surface $\gamma$ are implied by high gradients. In this case, it is common to minimize a functional which penalizes small gradients, as for example the Snakes energy introduced in [60]:

$$I_{KWT}(\gamma) = -\int_{\gamma} |\nabla f(\gamma)| dxdy + \alpha \text{Area}(\gamma) \tag{5.31}$$

Note that edge based energy functionals are usually minimized by active contour methods. They consist of evolving a small initial contour mesh towards the boundary of the object, which is a minimizer of the energy functional. An advantage of active contour methods is that the evolution converges to a minimizer close to the initial contour. However, since the evolution equations only contain local information, but shape prior is a global information, active contour methods cannot be used efficiently in this case. In order to retain control over local minimizers of (5.31), we use the term $\beta\text{Area}(\gamma)$ in (5.31) instead, thus forcing the contour closer to the initial contour.

In the presence of noisy data that e.g. arises in MRI or ultrasound imaging, minimization of both functionals (5.30) and (5.31) over $\gamma$ is not well-posed. Furthermore, if the object to segment is partially occluded, minimizers of (5.30) and (5.31) might not only segment the object, but also parts of the background, and knowledge about the true shape of the object to segment has to be considered.
Here we use the mean ball representation $\boldsymbol{\mu}$ of Equation (5.11) as a priori information and the Mahalanobis distance $d_\Sigma$ as defined in Subsection 5, formula (5.23) to measure the distance between two ball representations. We recall that with a given fixed ball representation $\boldsymbol{\mu}$, an equivalence class $[M]_{\sim_{ST}}$ of ball representations can be uniquely represented by tangent coordinates $P_{\boldsymbol{\mu}}(M) = (c_{\boldsymbol{\mu}}(M), r_{\boldsymbol{\mu}}(M))$, see (5.18). Tangent coordinates and a similarity transformation $(\rho, A, b) \in \mathbb{R}^{>0} \times SO(3) + \times \mathbb{R}^3$ uniquely determine

a skin surface $\gamma = \gamma(P_{\boldsymbol{\mu}}(M), f)$ as defined in Subsection 5, formula (5.28). We obtain the following functionals as a regularization of (5.30) resp. (5.31) for some $\beta > 0$:

$$I_\beta : \Sigma \times \mathrm{SO}(3) \times \mathbb{R}^3 \times \mathbb{R}^{\geq 0} \longrightarrow \mathbb{R},$$

$$(P_{\boldsymbol{\mu}}, f) \mapsto I_{SMS}(\gamma(P_{\boldsymbol{\mu}}, f)) + \beta d_\Sigma(0, P_{\boldsymbol{\mu}}) \tag{5.32}$$

$$(P_{\boldsymbol{\mu}}, f) \mapsto I_{KWT}(\gamma(P_{\boldsymbol{\mu}}, f)) + \beta d_\Sigma(0, P_{\boldsymbol{\mu}}) \tag{5.33}$$

Note that the regularization term actually measures the Mahalanobis distance between $M$ and $\boldsymbol{\mu}$, since $P_{\boldsymbol{\mu}}(\boldsymbol{\mu}) = 0$.

# 6. RESULTS

In the following section we outline the results for the application of our algorithm on two different test images: first, a synthetic dataset of torus voxel images was segmented using functional (5.32) in Section 5. As a second test case we study prostate segmentation in MRI datasets. Thereby we use functional (5.32), which is described in Section 5. Both functionals are minimized using the evolutionary algorithm CMA-ES (see [55, 65]). This choice is mainly motivated by the fact, that the calculation of the gradient for both functionals is tough, since it requires finding the derivative with respect to the skin surface, which is given by the coordinates of the medial ball representation. Thus analytical differentation is numerical differentiation should be used for this purpose. However, the skin surface meshing process we used for surface construction ([68]) does not guarantee that the resulting polyhedra always have the same structure or even the same number of vertices and faces, as it is the case for example by using spline based interpolation surfaces. Those irregularities in the resulting surface meshes complicate the numerical differentiation process and thus we decided to use gradient-free evolutionary minimization algorithms. We chose the evolutionary algorithm CMA-ES for the minimization of 5.32, because it has been successfully applied to a similar task in [32], where its superior behavior over two other common evolutionary minimization techniques was demonstrated. For more details on the minimization process, especially also concerning the choice of the minimization parameters, we also refer to [32].

## 6.1. Synthetic Examples

In this subsection we are concerned with the segmentation of a synthetically created and corrupted torus voxel image. by this example we demonstrate not only the influence of the shape prior on the segmentation result but in particular also the capability of the method to segment surfaces of arbitrary topology, i.elet@tokeneonedota torus in this case. We generated 15 binary training voxel images containing elliptic tori with differing radii and positions (see Figure 5.4). Using the software ITK-SNAP [119] we computed the according input meshes $\Omega_1, \ldots, \Omega_{15}$ which are the actual input data for the shape statistics.

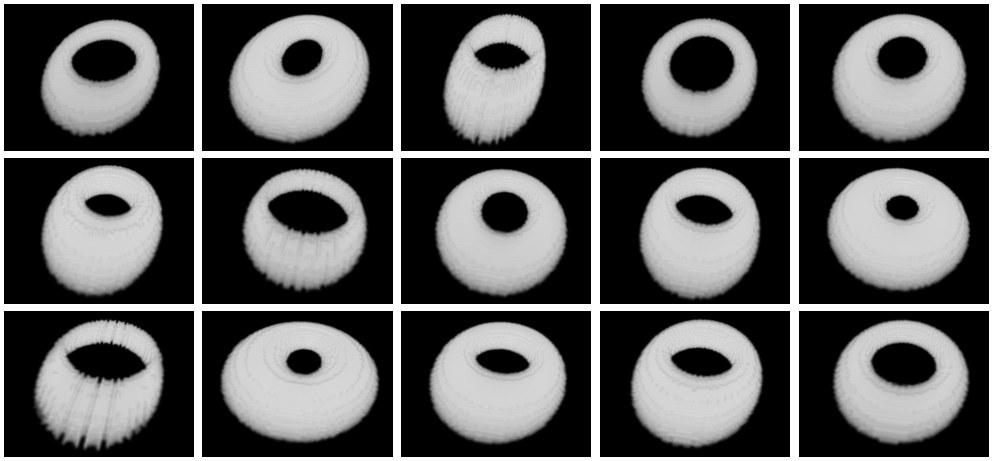


Figure 5.4: 15 different voxel images of tori used as training data

Using the surface meshes Scheme 5.1.1 has been used to compute a shape space and a Mahalanobis distance in it. This shape information was then used to segment a torus corrupted by a large black strip as shown in Figure 5.5(a). Because high contrasts occur in the images, we chose (5.32) as the segmentation functional, which we minimized using the CMA-ES. The segmentation algorithm was run with $k = 15$ balls. The shrink factor $s$ was included in the minimization process. As initial value for $s$ it proved suitable to take high values if long and thin structures were expected and smaller values in case of more compact objects. The results of the segmentation process with different values for the regularization parameter $\beta$ are shown in Figure 5.5(b)-(d). As expected, small values of $\beta$, i.elet@tokeneonedotsegmentation with low influence of the regularization, results in improper segmentation results (Figure 5.5(b)-(c)), while the right choice of $\beta$ yields a good segmentation, as seen in Figure 5.5(d). Note that the slight inaccuracies in the segmentation are a consequence of using skin surfaces as surface generation method, which often results in a surface of ball-like structure by nature.

## 6.2. Prostate Data

In this example, we are concerned with the segmentation of the prostates in T2-weighted MRI datasets which have been provided for the MICCAI Segmentation Challenge for Clinical Applications [1]. The dataset consisted of 15 training MRI images, which were accompanied by slice-by-slice expert segmentations. Additionally, one dataset was proposed for testing the segmentation algorithm. This dataset is shown in Figure 5.6(a). As can also be seen there, there are two main challenges for the segmentation algorithm: first, the MRI datasets are very thin, i.elet@tokeneonedotthe number of slices in z-direction is only 28 with the prostate ranging over only about 10 slices. This low resolution poses a general problem for 3D-segmentation algorithms due to rounding errors and leads to the fact, that state- of-the-art techniques for prostate segmentation are

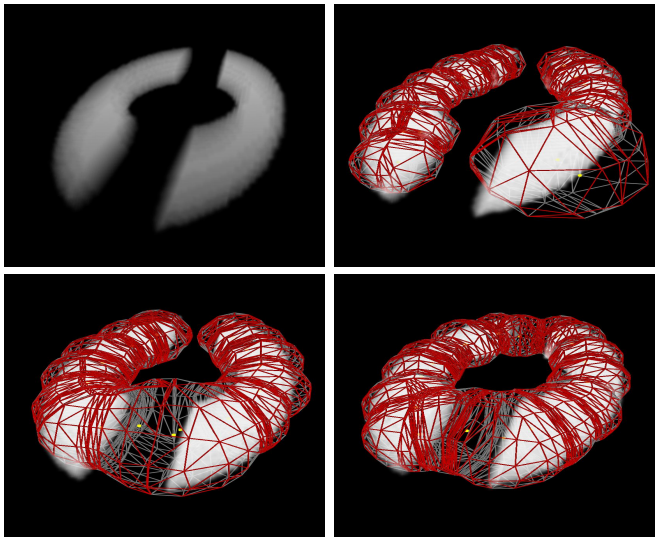


Figure 5.5: Segmentation results for the corrupted torus image for different values of the regularization parameter $\beta$. From upper left to lower right: (a) Voxel image of corrupted torus data. (b) Segmentation result with regularization parameter $\beta = 5 * 10^{-2}$ (c) $\beta = 10^{-2}$ (d) $\beta = 10^{-3}$.

often based on the combination of slice-by-slice 2D segmentations.

On the other hand the prostate in the MRI test dataset, which can be seen as the gray colored area in the middle of Figure 5.6(a), has a very low contrast difference to the other areas of the image. This problem leads to the fact, that a region based functional like (5.32) is not suitable in this case. Therefore, the edge based segmentation functional (5.33) was used for the segmentation process. For this purpose, we had to calculate the absolute gradient of the voxel image. To minimize the effect of noise, we first used anisotropic filtering to smooth the voxel image (Figure 5.6(b)). Afterwards the absolute gradient of the image was calculated using the morphological gradient (Figure 5.6(c)). Like in Subsection 5, we then again created 15 training meshes $\Omega_1, \ldots, \Omega_{15}$ out of the expert segmentations, which were used to compute shape space and Mahalanobis distance using Scheme 5.1.1 and the resulting functional combined with the shape prior was minimized using the CMA-ES. The result of this segmentation procedure is shown in Figure 5.7. As expected from the low input resolution and the complex image contrast structure, the functional needs a high value for the shape prior to yield a good segmentation result. Nevertheless the results are promising.

# 7. CONCLUSION

In this paper we presented a new approach for segmentation of 3D voxel images taking into account statistical shape information. The algorithm requires minimal user inter-

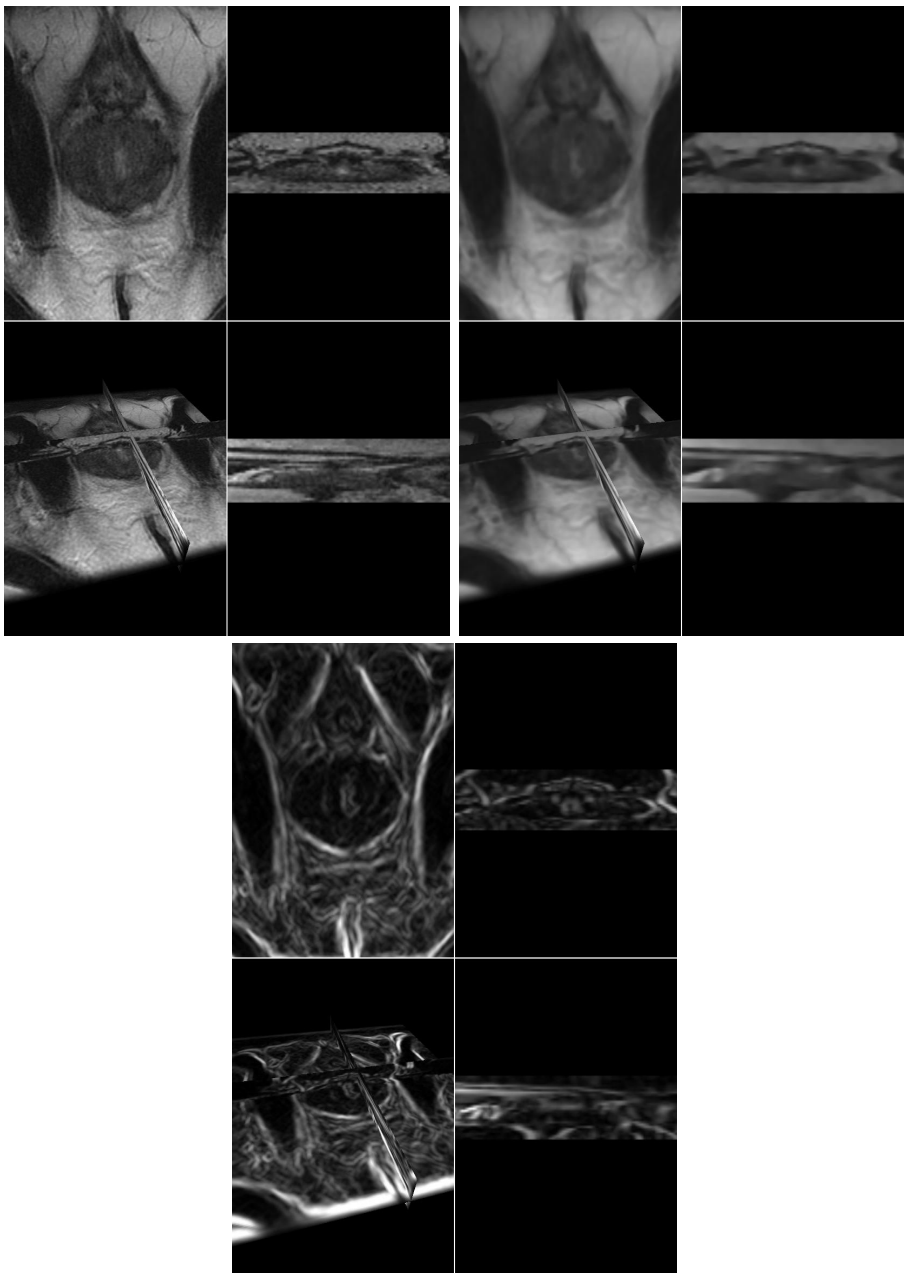


Figure 5.6: The figure shows (a) the original MRI test image $f$ on the upper left, (b) the MRI test image after smoothing using anisotropic filtering $f_\sigma$ on the upper right, and (c) the absolute value of the morphological gradient of the smoothed image $|\nabla f_\sigma|$ on the bottom. All images consist of four different views of the voxel image, i.elet@tokeneonedotfrom upper left to lower right: transversal, coronal, 3D, sagittal.

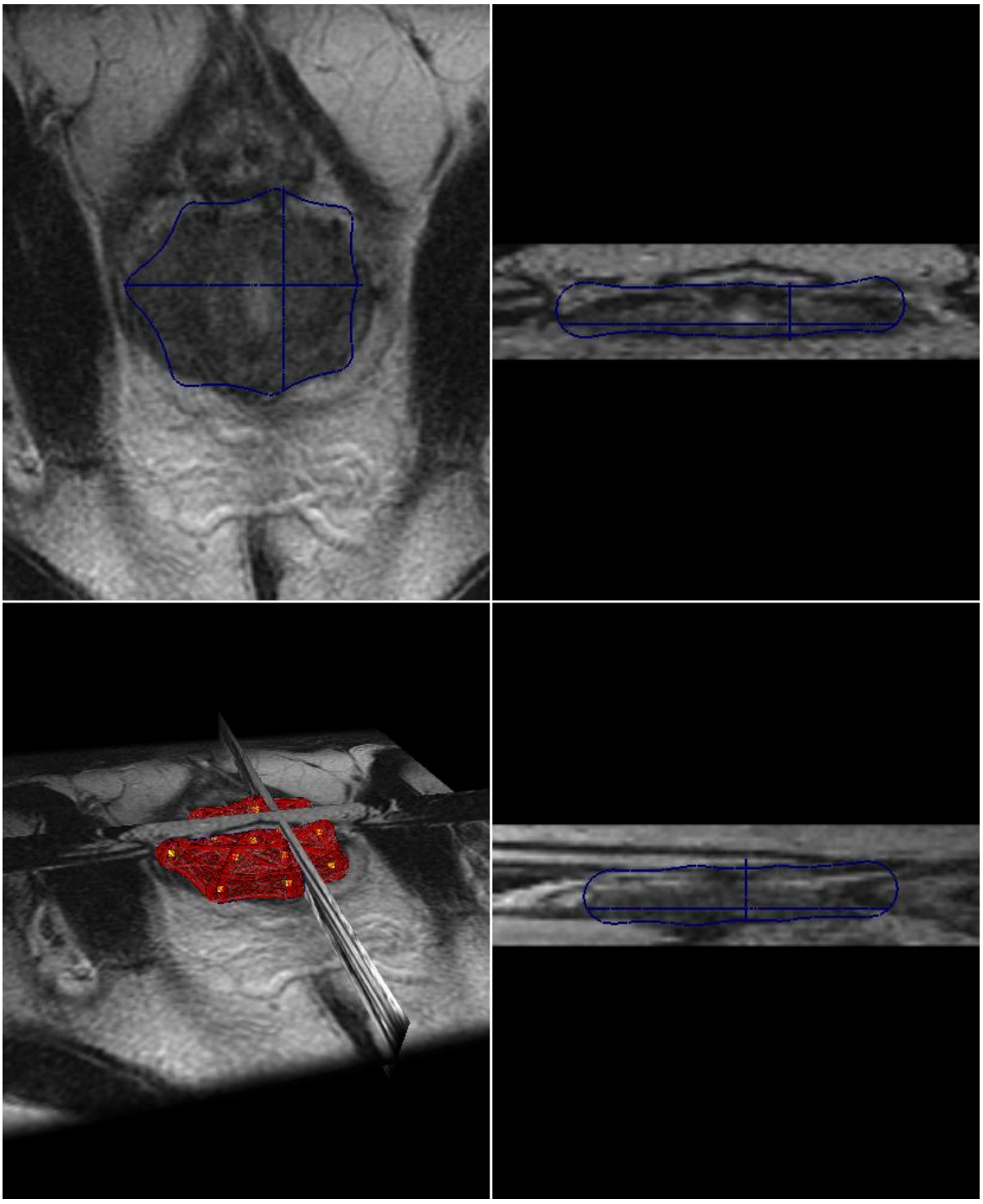


Figure 5.7: Segmentation result for the prostate in the MRI test image with regularization parameter $\beta = 0.5$. The image shows from upper left to lower right: transversal, coronal, 3D and sagital view of the original image $f$ and the segmentation result. The segmentation result in the 2D-views is visualized by the blue contours, while the resulting mesh surface and its medial ball centers is shown in 3D.

actions for segmentation which is achieved by implementing a pipeline of algorithms. The pipeline involves the automatic generation of training shapes represented as medial ball representations that were automatically generated from a given set of input meshes. Afterwards the training shapes are labeled, and a shape space based on Procrustes statistics is established. The resulting shape space and the Mahalanobis distance are used as a prior in region- and edge based segmentation algorithms.

We did a proof-of-concept for our algorithm by first evaluating its performance on a synthetic dataset of 15 randomly generated torus images. By utilizing a segmentation energy which includes statistical regularization using the PCA of the training data, we were able to segment a distorted torus image correctly. Additionally, this example shows the benefit of using skin surfaces as surface generation method for ball representations, which enables us to segment images of arbitrary topology (in contrast to for instance $B$-splines).

Furthermore, we applied our algorithm to the segmentation of MRI prostate data. Again, we used a statistically regularized segmentation functional supplied with 15 training shapes represented as medial balls which were automatically generated from the available expert segmentations. The automatic generation of medial balls from expert slice-by-slice segmentations is one of the advantages of our algorithm in contrast to the work of Pizer et allet@tokeneonedot(e.glet@tokeneonedot[91]), where expert segmentations have to be constructed by using a predefined medial ball model itself. The segmentation result demonstrates that it is in principle possible to automatically reconstruct shapes from MRI images with our method. We want to point out however, that a rigorous evaluation and comparison of the segmentation results for the prostate in MRI images or, more generally spoken, application in the field of medical image segmentation, requires specific fine-tuning of our general formulated algorithm to the specific application field, which is beyond the scope of our paper.

It was shown that the approximation of the medial axis by using only balls with no structure is sufficient in principle but also poses problems when dealing with complex applications. Thus, a future work will be to investigate ways of approximating also the structure of the medial surface. As a first idea, spline surface approximation techniques could be used to approximate the medial surface and interpolate over the resulting medial balls of the discrete medial axis transform. Furthermore, to obtain a better segmentation result for complex medical images one could investigate the use of energy functionals based not only on shape statistics but also image knowledge as e.glet@tokeneonedotoutlined in [50].

## Acknowledgement

This work has been supported by the Austrian Science Fund (FWF) within the research networks NFNs *Industrial Geometry*, Projects S09203 and S09205, and Photoacoustic Imaging in Biology and Medicine, Project S10505-N20.

# Appendix

The signed singular value decomposition.
Denote $S_{(i)}$ the $i$-th row, and $S^{(j)}$ the $j$-th column of a square matrix $S$. The signed singular value decomposition of $S$,

$$S = UDV^T \tag{5.34}$$

is obtained from the (usual) singular value decomposition $S = UDV^T$ in the following way

- if $\det(U) = \det(V) = 1$, then do nothing
- if $\det(U) = -\det(V) = 1$, replace the last column $V^{(d)}$ of $V$ by $-V^{(d)}$, and replace $D_{d,d}$ by $-D_{d,d}$.
- if $-\det(U) = \det(V) = 1$, replace the last column $U^{(d)}$ of $U$ by $-U^{(d)}$, and replace $D_{d,d}$ by $-D_{d,d}$.
- if $-\det(U) = -\det(V) = 1$, replace the last column $U^{(d)}$ of $U$ by $-U^{(d)}$ and $V^{(d)}$ by $-V^{(d)}$

Performing these manipulations, it is guaranteed that $\det(U) = \det(V) = 1$ and therefore $U$, $V \in SO(d)$.

Formulas for the maximization step in Algorithm (1).
Given $P$, the M-Step of Algorithm (1) consists in maximizing the functional (5.7),

$$\sum_{i,j=1}^{n} P_{j,i}(\log p_i + \log g_i(x_j, r_j))$$

with respect to $f$ and $p_1, \ldots, p_k$, . Since $p_1, \ldots, p_k$ and $f$ occur in different summands, maximization can be performed independently for each sum.

(1) Maximizing (5.7) over $p_1, \ldots, p_k$ subject to the restriction $\sum_{i=1}^{k} p_i = 1$ gives the condition

$$\sum_{j=1}^{k} P_{j,i} \frac{1}{p_i} = \lambda$$

with a Lagrange multiplier $\lambda \in \mathbb{R}$ for all $i$. Since $\sum_{i,j=1}^{k} P_{j,i} = k$ and $\sum_{i=1}^{k} p_i = 1$ it follows that $\lambda = k$ and therefore

$$p_i = \frac{1}{k} \sum_{j=1}^{k} P_{j,i}, \quad i = 1, \ldots, k.$$

(2) Maximization of (5.7) with respect to $\rho > 0$, $A \in SO(3)$ and $b \in \mathbb{R}^3$ is equivalent to minimization of

$$\sum_{i,j=1}^{k} P_{j,i} \left( ||x_j - \rho A y_i - b||^2 + \alpha (r_j - \rho s_i)^2 \right) \tag{5.35}$$

over $\rho > 0$, $A \in SO(3)$ and $b \in \mathbb{R}^3$.
We first consider the special case $\alpha = 0$ in (5.35), and then the general case $\alpha \geq 0$.

(a) The special case $\alpha = 0$.

We denote the weighted means of $x$ and $y$ by

$$\overline{x} = \frac{\sum\limits_{i,j=1}^{k} P_{j,i} x_j}{\sum\limits_{i,j=1}^{k} P_{j,i}} \quad \text{and} \quad \overline{y} = \frac{\sum\limits_{i,j=1}^{k} P_{j,i} y_i}{\sum\limits_{i,j=1}^{k} P_{j,i}} \, .$$

Moreover, let

$$S_{y,x} = \sum_{i,j=1}^{k} P_{j,i} (y_i - \overline{y})(x_j - \overline{x})^T .$$

Following the computations for the unweighted case in [81], we compute the signed singular value decomposition of $S_{y,x}$,

$$S_{y,x} = U D V^T \, .$$

Here, $U$, $V \in SO(3)$ and $D$ is diagonal with $D_{1,1} \geq D_{2,2} \geq |D_{3,3}|$. Then the solution of minimization problem (5.35) in case $\alpha = 0$ is given by

$$\rho = \frac{\text{Trace}(D)}{\text{Trace}(S_{x,x})} \tag{5.36}$$

$$A = U V^T \tag{5.37}$$

$$b = \overline{x} - A \overline{y} \tag{5.38}$$

(b) The general case $\alpha \geq 0$.

Since the additional term in case $\alpha \geq 0$ is independent of rotation and scaling, the formulas (5.37) and (5.38) for optimal translation and rotation remain the same in the general case. The scaling factor $\lambda$, computed by differentiation, computes to

$$\rho = \frac{\text{Trace}(D) + \alpha r^T P s}{\text{Trace}(S_{x,x}) + \alpha r^T P r} \tag{5.39}$$

# Bibliography


[1] Medical image computing and computer assisted intervention. http://www.miccai2009.org.

[2] J. Abhau. A robust and efficient method for topology adaptations in deformable models. In *VISAPP 2008: International conference on computer vision theory and applications*. Proceedings of VISAPP, 2008.

[3] J. Abhau, W. Hinterberger, and O. Scherzer. Segmenting surfaces of arbitrary topology: A two-step approach. In *Ultrasonic Imaging and Signal Processing*. Proceedings of SPIE – Volume 6437, 2007.

[4] J. Abhau and O. Scherzer. A combinatorial method for topology adaptations in 3d deformable models. *Int. J. Comput. Vision*, 87(3):304–315, 2010.

[5] D. Adalsteinsson and J.A. Sethian. A fast level set method for propagating interfaces. *J. Comput. Phys.*, 188(2):269–277, 1995.

[6] R.K. Ahuja, T.L. Magnanti, and J.B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993.

[7] O. Aichholzer, F. Aurenhammer, T. Hackl, M. Kornberger, B. M. Peternell, and H. Pottmann. Approximating boundary-triangulated objects with balls. In *Proc. 23rd European Workshop on Computational Geometry*, pages 130–133, Graz, 2007.

[8] O. Aichholzer, F. Aurenhammer, and B. Kornberger. Stable piecewise-linear approximations of 3d medial axes. Manuscript.

[9] O. Aichholzer, F. Aurenhammer, B. Kornberger, S. Plantinga, G. Rote, A. Sturm, and G. Vegter. Recovering structure from $r$-sampled objects. In *Proc. Symposium on Geometry Processing*, volume 28, pages 1349–1360, Berlin, 2009.

[10] O. Alexandrov and F. Santosa. A topology-preserving level set method for shape optimization. *J. Comput. Phys.*, 204(1):121–130, 2005.

[11] N. Amenta and M. Bern. Surface reconstruction by "voronoi" filtering. *Discrete & Computational Geometry*, 22:481–504, 1999.

[12] N. Amenta and R. Kolluri. Accurate and efficient unions of balls. In *Proc. 16th Ann. Symp. Computational Geometry*, pages 119–128, Hong Kong, 2000. ACM.

[13] B. Arnolds, H. Müller, D. Saupe, and T. Tolxdorff, editors. *Proceedings of 4th Freiburger Workshop* Digitale Bildverarbeitung in der Medizin*, Freiburg*. Universität Freiburg, Medizin-Technische Transferstelle 1994, Freiburg, 1996.

[14] D. Attali, J.D. Boissonnat, and H. Edelsbrunner. Stability and computation of medial axes - a state-of-the-art report. In T. Möller, B. Hamann, and R. Russell, editors, *Mathematical Foundations of Scientific Visualization, Computer Graphics, and Massive Data Exploration*. Springer, New York, 2009. Mathematics and Visualization.

[15] G. Bertrand. Simple points, topological numbers and geodesic neighborhoods in cubic grids. *Pattern Recognition Letters*, 15(10):1003–1011, 1994.

[16] S. Bischoff and L. Kobbelt. Snakes with topology control. *Visual Comp.*, 20:217–228, 2004.

[17] S. Bischoff and L. Kobbelt. Cartoon extraction based on anisotropic image classification. In *Vision, Modeling, and Visualization Proceedings*, pages 293–300, 2006.

[18] H. Blum and R.N. Nagel. Shape description using weighted symmetric axis features. *Pattern Recognition*, 10:167–180, 1978.

[19] J.-D. Boissonnat and M. Teillaud, editors. *Effective Computational Geometry for Curves and Surfaces*. Springer, 2006.

[20] F. L. Bookstein. *Morphometric tools for landmark data: geometry and biology*. Cambridge University Press, Cambridge, 1997.

[21] Y. Boykov and V. Kolmogorov. Computing geodesics and minimal surfaces via graph cuts. In *Ninth Conference on Computer Vision*, volume 1, pages 26–33. IEEE, 2003.

[22] J. Bredno, T.M. Lehmann, and K. Spitzer. General discrete contour model in two, three, and four dimensions for topology-adaptive multichannel segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(5):550–563, 2003.

[23] X. Bresson, P. Vandergheynst, and J.P. Thiran. A variational model for object segmentation using boundary information and shape prior driven by the mumford-shah functional. *Int. J. Comput. Vision*, 28(2):145–162, 2006.

[24] J.F. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, PAMI-8:679–697, 1986.

[25] V. Caselles, F. Catté, T. Coll, and F. Dibos. A geometric model for active contours in image processing. *Numer. Math.*, 66(1):1–31, 1993.

[26] V. Caselles, R. Kimmel, and G. Sapiro. Geodesic active contours. *Int. J. Comput. Vision*, 22(1):61–79, 1997.

[27] T. Chan and L. Vese. Active contours without edges. *IEEE Trans. Image Process.*, 10(2):266–277, 2001.

[28] Q. Chen, Z.M. Zhou, M. Tang, P.A. Heng, and D.S. Xia. Shape statistics variational approach for the outer contour segmentation of left ventricle mr images. *IEEE Trans. Inf. Technol. Biomed.*, 10(3):588–597, 2006.

[29] Y. Chen and G. Medioni. Description of complex objects from multiple range images using an inflating balloon model. *Comput. Vision Image Understanding*, 61, No 3:325–334, 1995.

[30] Y. Chen, H.D. Tagare, S. Thiruvenkadam, F. Huang, D. Wilson, K.S. Gopinath, R.W. Briggs, and E.A. Geiser. Using prior shapes in geometric active contours in a variational framework. *Int. J. Comput. Vision*, 50:315–328, 2002.

[31] S. Colutto, F. Fruehauf, M. Fuchs, and O. Scherzer. The CMA-ES on Riemannian manifolds to reconstruct shapes in 3D voxel images. Reports of FSP S092 - "Industrial Geometry" 70, University of Innsbruck, Austria, 2008. submitted.

[32] S. Colutto, F. Frühauf, M Fuchs, and O. Scherzer. The CMA-ES on Riemannian manifolds to reconstruct shapes in 3D voxel images. *IEEE Transactions on Evolutionary Computation*, 2009. to appear.

[33] T.F. Cootes, C.J. Taylor, D.H. Cooper, and J. Graham. Active shape models - their training and application. *Comput. Vision Image Understanding*, 61(1):38–59, 1995.

[34] M. G. Crandall, H. Ishii, and P.-L. Lions. User's guide to viscosity solutions of second order partial differential equations. *Bull. Amer. Math. Soc.*, 27(1):1–67, 1992.

[35] D. Cremers, T. Kohlberger, and C. Schnoerr. Shape statistics in kernel space for variational image segmentation. *Pattern Recognition*, 36(9):1929–1943, 2003.

[36] D. Cremers, F. Tischhäuser, J. Weickert, and Ch. Schnörr. Diffusion snakes: Introducing statistical shape knowledge into the Mumford-Shah functional. *Int. J. Comput. Vision*, 50:295–313, 2002.

[37] H. Delingette. Adaptive and deformable models based on simplex meshes. In *IEEE Workshop of Non-Rigid and Articulated Objects*. IEEE Computer Society Press, 1994.

[38] H. Delingette. General object reconstruction based on simplex meshes. *Int. J. Comput. Vision*, 32:111–142, 1999.

[39] T. K. Dey, H. Edelsbrunner, and S. Guha. Computational topology. In *Advances in Discrete and Computational Geometry (Contemporary mathematics 223)*, pages 109–143. American Mathematical Society, 1999.

[40] I.L. Dryden and K.V. Mardia. *Statistical Shape Analysis*. John Wiley, Chichester, 1998.

[41] H. Edelsbrunner. Deformable smooth surface design. *Discrete Comp. Geom.*, 21:87–115, 1999.

[42] H. Edelsbrunner and E.P. Mücke. Simulation of simplicity: A technique to cope with degenerate cases in geometric algorithms. *ACM Trans. Graph.*, 9(1):66–104, 1990.

[43] N. El-Zehiry, S. Xu, P. Sahoo, and A. Elmaghraby. Graph cut optimization for the mumford-shah model. In *Proceedings of the seventh IASTED international conference, visualization, imaging and image processing*. 2007.

[44] W. Fang and K.L. Chan. Incorporating shape prior into geodesic active contours for detecting partially occluded object. *Pattern Recognition*, 40(7):2163–2172, 2007.

[45] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *Int. J. Comput. Vision*, 59(2):167–181, 2004.

[46] P.T. Fletcher, S. Joshi, C. Ju, and S.M. Pizer. Principal geodesic analysis for the study of nonliner statistics of shape. *IEEE Trans. Med. Imag.*, 23:995–1005, 2004.

[47] P.T. Fletcher, S. Joshi, C. Lu, and S.M. Pizer. Gaussian distributions on lie groups and their application to statistical shape analysis. *Lecture Notes in Computer Science*, 2732:450–462, 2003. IPMI 2003.

[48] P.T. Fletcher, S.M. Pizer, and S.C. Joshi. Shape variation of medial axis representations via principal geodesic analysis on symmetric spaces. In *Statistics and analysis of shapes*, Model. Simul. Sci. Eng. Technol., pages 29–59. Birkhäuser Boston, Boston, MA, 2006.

[49] D.S. Fritsch, S.M. Pizer, L. Yu, V. Johnson, and E.L. Chaney. Localization and segmentation of medical image objects using deformable shape loci. *Lecture Notes in Computer Science*, 1230:127–140, 1997. IPMI 1997.

[50] M. Fuchs and S. Gerber. Variational shape detection in microscope images based on joint shape and image feature statistics. In *CVPR Workshops 2008. IEEE Computer Society Conference on*, pages 1–8, 2008.

[51] M. Gastaud, M. Barlaut, and G. Aubert. Combining shape prior and statistical features for active contour segmentation. *IEEE Trans. Circuits and Systems*, 14(5):726–734, 2004.

[52] Y. Giga. *Surface evolution equations*, volume 99 of *Monographs in Mathematics*. Birkhäuser Verlag, Basel, 2006. A level set approach.

[53] R. Goldenberg, R. Kimmel, E. Rivlin, and M. Rudzsky. Cortex segmentation: a fast variational geometric approach. *IEEE Trans. Med. Imag.*, 21(12):1544–1551, 2002.

[54] X. Han, C. Xu, and J.L. Prince. A topology preserving level set method for geometric deformable models. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(6):755–768, 2003.

[55] N. Hansen, S. D. Müller, and P. Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). *Evolutionary Computation*, 11(1):1–18, 2003.

[56] A. Hatcher. *Algebraic Topology*. Cambridge University Press, 2002.

[57] M. Holtzman-Gazit, R. Kimmel, N. Peled, and D. Goldsher. Segmentation of thin structures in volumetric medical images. *IEEE Trans. Image Process.*, 15(2):354–363, 2006.

[58] S. Joshi, S. Pizer, P.T. Fletcher, A. Thall, and G. Tracton. Multi-scale 3-d deformable model segmentation based on medical description. In *Proc. International Conference on Information Processing in Medical Imaging (IPMI)*, pages 64–77, 2001.

[59] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(7):881–892, 2002.

[60] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *Int. J. Comput. Vision*, 1(4):321–331, 1987.

[61] M. Kazhdan. Reconstruction of solid models from oriented point sets. In *Symposium on Geometry Processing*, pages 73–82, 2005.

[62] D.G. Kendall. Shape manifolds, procrustean metrics and complex projective spaces. *Bull. Lond. Math. Soc.*, 16:81–121, 1984.

[63] J. T. Kent, K.V. Mardia, and C.C. Taylor. Matching problems for unlabelled configurations. *Bioinf. Images Wavelets*, pages 33–36, 2004.

[64] J.T. Kent, K.V. Mardia, and C.C. Taylor. Matching unlabelled configurations and protein bioinformatics. *to appear*.

[65] S. Kern and N. Hansen. Evaluating the cma evolution strategy on multimodal test functions. In *Eigth International Conference on Parallel Problem Solving from Nature PPSN VIII*, volume 3242, pages 282–291. Springer, 2004.

[66] L. Kobbelt, S. Campagna, J. Vorsatz, and H. P. Seidel. Interactive multiresolution modeling on arbitrary meshes. In *Computer Graphics (SIGGRAPH 98 Proceedings)*. SIGGRAPH, 1998.

[67] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *IEEE Trans. Pattern Anal. Mach. Intell.*, 26:147–159, 2004.

[68] N. Kruithof and G. Vegter. Meshing skin surfaces with certified topology. *Comput. Geom.*, 36:166–182, 2007.

[69] G. Kuehne and J. Weickert. *Fast methods for implicit active contour models*, pages 43–57. Springer Verlag, 2003.

[70] J. O. Lachaud and A. Montanvert. Deformable meshes with automated topology changes for coarse-to-fine three-dimensional surface extraction. *J. Med. Image Anal.*, 3, No 2:187–207, 1999.

[71] J. O. Lachaud and B. Taton. Deformable model with a complexity independent from image resolution. *Comput. Vision Image Understanding*, 99(3):453–475, 2005.

[72] J.O. Lachaud and B. Taton. Deformable model with adaptive mesh and automated topology changes. In *Proceedings of 4th International Conference on 3-D Digital Imaging and Modeling (3DIM'2003)*, 2003.

[73] J.O. Lachaud and B. Taton. Resolution independent deformable model. In *International Conference on Pattern Recognition (ICPR'2004)*, pages 237–240, 2004.

[74] S. Lang. *Algebra*. Springer, graduate texts in mathematics edition, 2002.

[75] S. Lang. *Algebra*. Springer, graduate texts in mathematics edition, 2005.

[76] C. Le Guyader and L. Vese. Self-repelling snakes for topology-preserving segmentation models. Preprint, UCLA C.A.M. Report 07-20, 2007.

[77] F. Leitner and P. Cinquin. Complex topology 3d-objects segmentation. In *SPIE Conference on Model Based Vision Development and Tools*, pages 16–26, 1992.

[78] M.E. Leventon, W.E.L. Grimson, and O. Faugeras. Statistical shape influence in geodesic active contours. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR-00)*, pages 316–323, Los Alamitos, 2000. IEEE.

[79] W. E. Lorensen and H. E. Cline. Marching cubes: a high resolution 3d surface construction algorithm. In *Computer Graphics*, volume 4, 1987.

[80] C. Lu, S.M. Pizer, and S. Joshi. A markov random field approach to multi-scale shape analysis. *Lecture Notes in Computer Science*, 2695:416–431, 2003. Scale Space Methods in Computer Vision.

[81] K.V. Mardia, J.T. Kent, and J.M. Bibby. *Multivariate Analysis*. Academic Press, London, 1977.

[82] W.S. Massey. *A Basic Course in Algebraic Topology*. Springer Verlag, 1991.

[83] T. McInerney and D. Terzopoulos. A finite element model for 3d shape reconstruction and nonrigid motion tracking. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 518–523, 1993.

[84] T. McInerney and D. Terzopoulos. T-snakes: Topology adaptive snakes. *Med. Image Anal.*, 4(2):73–91, 2000.

[85] G. McLachlan and T. Krishnan. *The EM algorithm and extensions*, volume 2. Wiley-Interscience, Hoboken, NJ, 2008.

[86] T. Moller. A fast triangle-triangle intersection test. *J. Graph. Tools*, 2/2:25–30, 1997.

[87] D. Mumford and J. Shah. Optimal approximations by piecewise smooth functions and associated variational problems. *Comm. Pure Appl. Math.*, 42(5):577–685, 1989.

[88] S. Osher and N. Paragios, editors. *Geometric Level Set Methods in Imaging, Vision, and Graphics*. Springer Verlag, New York, 2003.

[89] S. Osher and J. A. Sethian. Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton–Jacobi formulations. *J. Comput. Phys.*, 79(1):12–49, 1988.

[90] PARI. *PARI/GP, version* `2.1.7`. The PARI Group, Bordeaux, 2005. http://pari.math.u-bordeaux.fr/.

[91] S.M. Pizer, P.T. Fletcher, S. Joshi, A. Thall, J.Z. Chen, Y. Fridman, D.S. Fritsch, A.G. Gash, J.M. Glotzer, M.R. Jiroutek, C. Lu, K.E. Muller, G. Tracton, P. Yushkevich, and E.L. Chaney. Deformable m-reps for 3d medical image segmentation. *Int. J. Comput. Vision*, 55(2):85–106, 2003.

[92] S.M. Pizer, A.L. Thall, and D.T Chen. M-reps: A new object representation for graphics. Technical Report TR99-030, Department of Computer Science, University of North Carolina — Chapel Hill, 1999.

[93] J. P. Pons and J. D. Boissonnat. Delaunay deformable models: Topology-adaptive meshes based on the restricted delaunay triangulation. In *Proc. Conf. Computer Vision and Pattern Recognition (CVPR '07)*. IEEE Computer Society Press, 2007.

[94] M. A. Puso and T. A. Laursen. A 3-d contact smoothing algorithm method using gregory patches. *Numer. Meth. in Engineering*, 2002.

[95] K. R. Rocha, G. Sundaramoorthi, A. J. Yezzi, and J. L. Prince. 3d topology preserving flows for viewpoint-based cortical unfolding. *Int. J. Comput. Vision*, 2009.

[96] J. J. Rotman. *An Introduction to Algebraic Topology*. Springer, New York, 1988.

[97] M. Rousson and N. Paragios. Shape priors for level set representations. In *European Conference in Computer Vision (ECCV)*, pages 78–93, 2002.

[98] M. Rousson and N. Paragios. Prior knowledge, level set representations & visual grouping. *Int. J. Comput. Vision*, 76(3):231–243, 2007.

[99] O. Scherzer, M. Grasmair, H. Grossauer, M. Haltmeier, and F. Lenzen. *Variational Methods in Imaging*, volume 167 of *Applied Mathematical Sciences*. Springer, New York, 2009.

[100] F. Segonne. Active contours under topology control genus preserving level sets. *Int. J. Comput. Vision*, 79:107–117, 2008.

[101] J.A. Sethian. A marching level set method for monotonically advancing fronts. *Proc. Nat. Acad. Sci. U.S.A.*, 93(4):1591–1595, 1996.

[102] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):888–905, 2000.

[103] C. G. Small. *The Statistical Theory of Shape*. Springer Series in Statistics. Springer, 1996.

[104] D. Stalling and H.-C. Hege. Intelligent scissors for medical image segmentation. In *[13]*, pages 32–36, 1996.

[105] J.M. Sullivan. Curvature measures for discrete surfaces, 2002. available from http://torus.math.uiuc.edu/jms/Papers/.

[106] G. Sundaramoorthi and A. Yezzi. More-than-topology-preserving flows for active contours and polygons. In *Tenth IEEE International Conference on Computer Vision (ICCV'05)*, pages 1276–21283, 2005.

[107] S. Suri. Bipartite matching and the hungarian method. *www.cse.ust.hk/ golin/COMP572/Notes/Matching.pdf*.

[108] R. Szeliski, D. Tonnesen, and D. Terzopoulos. Modeling surfaces of arbitrary topology with dynamic particles. In *Proc. Conf. Computer Vision and Pattern Recognition (CVPR'93)*. IEEE Computer Society Press, 1993.

[109] G. Taubin. A signal processing approach to fair surface design. In *Computer Graphics (SIGGRAPH 95 Proceedings)*, pages 351–358, 1985.

[110] D. Terzopoulos and M. Vasilescu. Sampling and reconstruction with adaptive meshes. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 70–75, 1991.

[111] M. Teschner, B. Heidelberger, M. Mueller, D. Pomeranets, and M. Gross. Optimized spatial hashing for collision detection of deformable objects. In *Proceedings of Vision, Modeling, Visualization*, pages 47–54, 2003.

[112] M. Teschner, S. Kimmerle, B. Heidelberger, G. Zachmann, L. Raghupathi, A. Fuhrmann, M.P. Cani, F. Faure, N. Magnenat-Thalmann, W. Strasser, and P. Volino. Collision detection for deformable objects. *Comput. Graphics Forum*, 24:61–81, 2005.

[113] A. Tsai, A. Yezzi, C. Tempany, D. Tucker, A. Fan, W.E.L. Grimson, and A. Willsky. A shape-based approach to the segmentation of medical imagery using level sets. *IEEE Trans. Med. Imag.*, 22(2):137–154, 2003.

[114] J. Weickert and G. Kuehne. Fast methods for implicit active contour models. In *[88]*, pages 43–57. Springer, 2003.

[115] J. Weickert, B.M. ter Haar Romeny, and M.A. Viergever. Efficient and reliable schemes for nonlinear diffusion filtering. *IEEE Trans. Image Process.*, 7(3):398–410, 1998.

[116] A. Witkin, M. Kass, and D. Terzopoulos. Snakes: Active contour models. *Int. J. Comput. Vision*, 1, No 4:321–331, 1987.

[117] Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. 15(11):1101–1113, 1993.

[118] P. Yan and A.A. Kassim. Segmentation of volumetric mra images by using capillary active contour. *Med. Image Anal.*, 10:317–329, 2006.

[119] P.A. Yushkevich, J. Piven, C. Hazlett, H.and G. Smith, R. Ho, S., J.C. Gee, and G. Gerig. User-guided 3D active contour segmentation of anatomical structures: Significantly improved efficiency and reliability. *Neuroimage*, 31(3):1116–1128, 2006.

[120] X. Zeng, W. Chen, and Q. Peng. Efficiently solving the piecewise constant mumford-shah model using graph cuts. Technical report, Department of Computer Science, Zheijang University, P.R. China, 2006.

# Curriculum Vitae

## Personal Details


| | |
|---|---|
| Name | Jochen Abhau |
| Address | University of Innsbruck<br>Technikerstraße 21a<br>6020 Innsbruck, Austria |
| e-mail | jochen.abhau@uibk.ac.at |
| Date of Birth | 17/1/1978 |
| Place of Birth | Mülheim an der Ruhr |
| Nationality | German |


## Education

| | |
|---|---|
| 1988–1997 | Grammar School Broich, Mülheim an der Ruhr, Germany |
| 1998–2000 | Studies of Applied Mathematics and Business Administration<br>University Trier, Germany |
| 2000-2005 | Studies of Mathematics and Business Administration,<br>Bonn, Germany |
| 2005 | MSc in Mathematics. Diploma thesis: Die Homologie<br>von Modulräumen Riemannscher Flächen - Berechnungen für $g < 3$<br>Supervisor: Univ. Prof. Dr. Carl-Friedrich Bödigheimer |
| since 2005 | Dissertation: Geometrical and<br>Statistical Methods for Segmentation<br>of Topologically Complex Objects in 3D<br>Supervisor: Univ. Prof. Dr. Otmar Scherzer |

## Professional and Research Experience

| | |
|---|---|
| 1997–1998 | Civil Service, Theodor-Fliedner Stiftung,<br>Mülheim an der Ruhr, Germany |
| Oct 1999–Feb 2007 | Tutorials and courses in Mathematics at the Universities<br>of Trier, Bonn and Innsbruck |
| Aug 2000–Nov 2000 | Research project: Option Pricing with New Optimization<br>Methods, University of Trier |
| Dec 2001–Feb 2002 | Internship at Deka Investment Bank,<br>Quantitative Products, Frankfurt, Germany |
| Sep 2005–Mar 2009 | Research assistant at Infmath Imaging,<br>Department of Mathematics, University Innsbruck |
| Apr 2009–Sep 2009 | Research assistant at Industrial Mathematics<br>Competence Center (IMCC), Linz |
| Oct 2009–Jan 2010 | Research assistant at Computational Science Center,<br>University of Vienna |

## Publications

| | |
|---|---|
| 2005 | JA.<br>Die Homologie von Modulräumen Riemannscher<br>Flächen - Berechnungen für $g < 3$.<br>Diploma thesis, University Bonn |

| | |
|---|---|
| 2007 | JA, W. Hinterberger and O. Scherzer.<br>Segmenting surfaces of arbitrary topology: A two-step approach.<br>In SPIE Medical Imaging 2007: Ultrasonic Imaging and Signal Processing, Stanislav Y. Emelianov and Stephen A. McAleavey, Editors, Vol. 6513 |
| 2008 | JA, R. Ehrenfried and C.-F. Bödigheimer.<br>Homology Computations for Mapping Class Groups and Moduli Spaces of Surfaces with Boundary.<br>Heiner Zieschang Gedenkschrift,<br>Geometry and Topology Monographs 14, 1-25 |
| 2008 | JA.<br>A robust and efficient method for topology adaptations in deformable models. In VISAPP 2008: International conference on computer vision theory and applications.<br>Proceedings of VISAPP, 375-382 |
| 2008 | JA and O. Scherzer.<br>An efficient topology adaptation system for<br>In SPIE Medical Imaging 2008: Image Processing,<br>Joseph M. Reinhardt and Josien P. W. Pluim, Editors, Vol.6914 |
| 2009 | JA, Z. Belhachmi and O. Scherzer.<br>On a decomposition model for optical flow.<br>In 7th Conference on Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR09),<br>126139, Springer |
| 2010 | JA and O. Scherzer.<br>A combinatorial method for topology adaptations in 3D deformable models. International Journal of Computer Vision, 87(3), 304-315 |
| submitted | JA, O. Aichholzer, S. Colutto, B. Kornberger and O. Scherzer.<br>Medial axes shape spaces and segmentation of 3D voxel data.<br>submitted to EJAM |

## Conferences and Workshops

| | |
|---|---|
| Sep 25-Oct 1 2005 | Low-Dimensional Manifolds,<br>Oberwolfach, Germany |
| Nov 23-25 2005 | 1st FSP Seminar Industrial Geometry<br>Graz, Austria |
| Jun 19-22 2006 | 2nd FSP Seminar Industrial Geometry<br>Strobl, Austria |
| Sep 1-5 2006 | 3rd FSP Seminar Industrial Geometry, Variational, PDE and Level Set Methods, Obergurgl, Austria |
| Nov 27-29 2006 | FSP Workshop Industrial Geometry<br>Vorau, Austria |
| Feb 17-22 2007 | SPIE Medical Imaging,<br>San Diego, USA |

| | |
|---|---|
| Sep 15-18 2007 | Workshop on Polyhedral Surfaces and Industrial Applications Strobl, Austria |
| Jan 22-25 2008 | Computer Vision Theory and Applications, Funchal, Portugal |
| Feb 17-21 2008 | SPIE Medical Imaging, San Diego, USA |
| Apr 15-16 2008 | 1st NFN Seminar Photoacoustic Imaging Strobl, Austria |
| Mar 26-28 2008 | 5th FSP Seminar Industrial Geometry Strobl, Austria |
| Jun 1-7 2008 | Summer school in shape analysis and directional data, Pietracamela, Italy |
| Sep 24-26 2008 | 2nd NFN Seminar Photoacoustic Imaging Kitzeck, Austria |
| Apr 2-4 2009 | Analytic and computational methods in multi-scale modelling and applications, Metz, France |
| Apr 15-17 2009 | 6th NRN Seminar Industrial Geometry Strobl, Austria |
| Aug 24-27 2009 | 7th Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition, Bonn, Germany |