SEGMENTATION ON SURFACES BY THE CLOSEST POINT METHOD

by

Li (Luke) Tian

B.Sc., University of Alberta, 2006

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF APPLIED SCIENCE
in the School
of
Mathematics

APPROVAL

Name:	Li (Luke) Tian
Degree:	Master of Applied Science
Title of Thesis:	Segmentation on Surfaces by the Closest Point Method

Examining Committee: Dr. Paul Tupper
Chair

Dr. Steven Ruuth, Senior Supervisor

Dr. Manfred Trummer, Supervisor

Dr. Bob Russell, Examiner

Date Approved: November 30, 2009

Abstract

This thesis proposes a method to detect objects and patterns in textures on general surfaces. The approach applies the Chan-Vese variational model for active contours without edges to the problem of segmentation of scalar surface data. This leads to gradient descent equations which are level set equations on surfaces. These equations are evolved using the Closest Point Method, which is a recent technique for solving partial differential equations (PDEs) on surfaces. The final algorithm has a particularly simple form: it merely alternates a time step of the usual Chan-Vese model in a small 3D neighborhood of the surface with an interpolation step. The method can treat very general surfaces since it uses a closest point function to represent the underlying surface.

The original Chan-Vese active contours without edges model is intended for segmentation on a given 2D image based on curve evolution by minimizing an energy. The stopping term does not depend on the gradient of the image, as in the classical active contour models, but is instead related to a particular segmentation of the image [6]. Moreover, the Chan-Vese model is not just applicable or limited to 2D image segmentation. It can be further extended to 3D region segmentation and the segmentation of scalar data on surfaces defined in 3D. In order to apply the Chan-Vese model to the segmentation on surface problem, a PDE defining a flow on the surface in terms of intrinsic in-surface differential operators [13] must be solved.

There are various ways to solve this PDE, such as parameterization, and embedding methods using level set surface representation, etc; however, these methods have their own limitations. For example, some surfaces might be very challenging to parameterize; embedding methods with level sets can introduce artificial boundary conditions. To avoid these limitations, we use the Closest Point Method [21] for solving this surface PDE.

Various numerical experiments are presented. These include segmentation on a smooth surface, a non-smooth surface, an open surface, and a triangulated surface. We also present an efficiency improvement to accelerate convergence to a steady-state in Closest Point Method calculations.

Acknowledgments

The research behind this thesis was partially supported financially by a Simon Fraser University Graduate Fellowship and an NSERC Canada Grant.

I am thankful for all the technical and financial support given by my supervisor Dr. Steve Ruuth. I also thank Dr. Colin Macdonald for his help.

Finally I would like to thank all the instructors of my graduate study for their support.

Contents

List of Figures

Chapter 1

Introduction

1.1 Background

Segmentation of images has been studied extensively by researchers around the world. The segmentation of images is a topic with a great number of applications. For example, segmentation algorithms might be used to locate tumors in an ultrasonic image (see Figure 1) or to identify a building within an image (see Figure 2). Segmentation of objects and patterns in textures on *curved* surfaces (e.g., Figure 1.3) is a subject of recent and increasing interest. For example, Spira and Kimmel [22] segment out images painted on parametrically-defined manifolds using the geodesic active contour model for segmentation. Hara et al. [7] segment images on polar coordinate meshes using the Chan–Vese model. Applications of their methods arise in the analysis of earth data such as topography and remote sensing imagery. Bogdanova et al. [3] carry out segmentation via active contours for omnidirectional images defined on spherical, hyperbolical and parabolical shapes (such imagery arises using catadioptric cameras). Segmentation on surfaces via geometric active contours was also considered by Krueger et al. in [12]. Their work considers segmentation by both texture and surface geometry and leads to methods of interest in 3D human face feature segmentation.

1.2 Analysis of the Problem

A common theme to all these works is that they solve some PDE-based segmentation model on a surface of interest. The representation of the (static) surface is a central feature of each approach. For example, a parameterization might be used [22, 7, 3] but this has the deficiency of introducing distortions and singularities into the method through the parameterization. Indeed, designing a

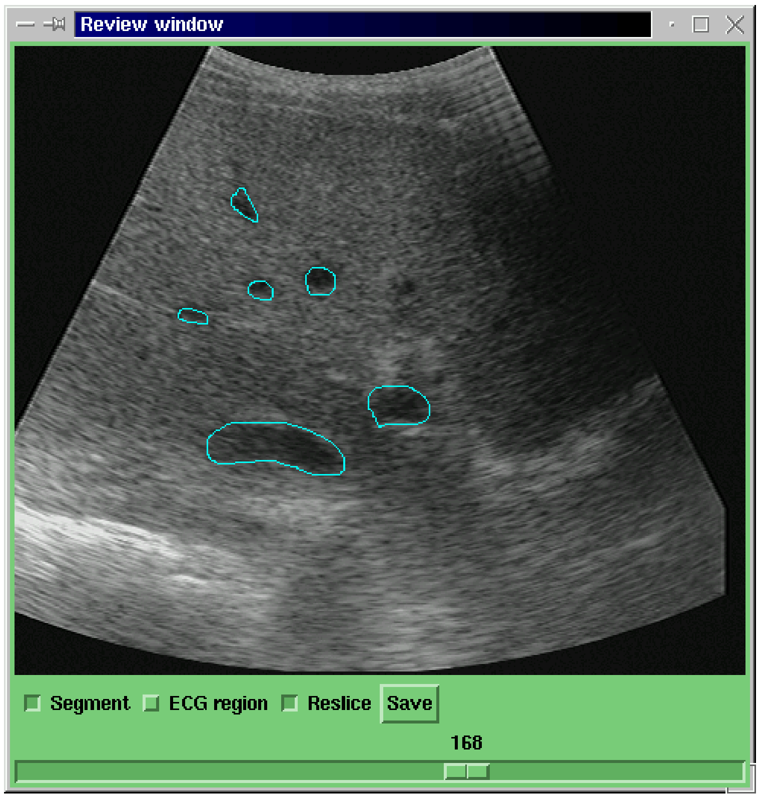

Figure 1.1: Segmentation can be applied to ultrasonic imaging in the medical field [20].

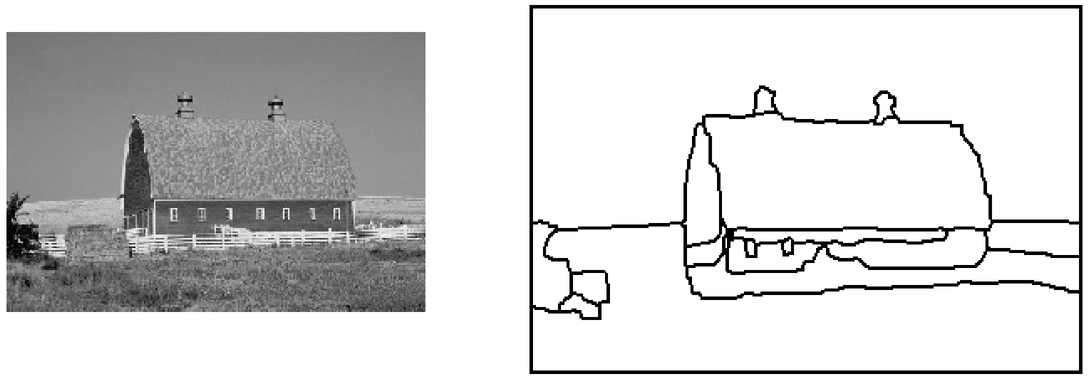

Figure 1.2: The segmentation on a 2D image of a barn by data driven Markov chain Monte Carlo [23].

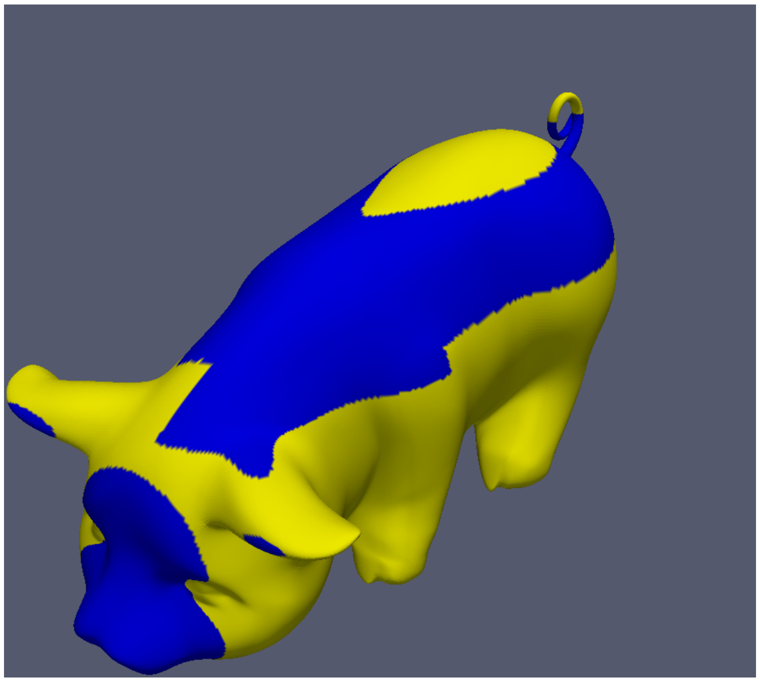
Figure 1.3: The objective of this thesis is to automatically segment a surface into two regions: in this case, to separate the blue patches from the rest of the pig.

good parameterization can be a considerable challenge for many surfaces. Alternatively, a level set representation of the underlying surface may be considered [12]. Algorithms based on level set representations have their own limitations [21]. Such methods solve a PDE in a 3D neighborhood of the surface and require the introduction of artificial boundary conditions at the boundary of the computational band. They also use non-standard PDEs in the embedding space, i.e., degenerate PDEs involving projection operators. Finally, level set representations do not give an obvious way to treat open surfaces, or any surface which lacks a clearly defined inside and outside.

For these reasons we are motivated to use the Closest Point Method [21, 14, 15] for solving the problem of segmentation on surfaces. The Closest Point Method uses a closest point representation of the surface, and therefore does not require the design of a parameterization nor does it require the surface to be closed or even have an orientation. The method uses entirely standard methods in a small 3D neighborhood of the surface in combination with an interpolation step. This is particularly attractive in practice since it enables us to use existing algorithms for 3D region segmentation to segment out shapes on 2D surfaces.

1.3 Thesis Overview

All the segmentation results are obtained using the Chan-Vese active contours without edges model [6], which we review in Chapter 3. Chapter 4 applies the Closest Point Method, a method which is discussed in Chapter 2, to the Chan-Vese model.

Chapter 2

The Closest Point Method

2.1 Introduction

The Closest Point Method is a new technique for solving PDEs on surfaces. For example, the Closest Point Method could be used to compute a numerical solution of a PDE on the surface of a sphere, square, torus, Mobius strip or some triangulated surface.

2.2 Background

2.2.1 Partial Differential Equations on Surfaces

Partial differential equations play an essential role in the natural sciences and other areas. In many cases, the problems of interest occur on surfaces rather than in $\mathbb{R}^2$ or $\mathbb{R}^3$. The solution of PDEs on surfaces has received growing interest over the past few years. Recent computer science applications include texture mapping, computer vision and digital image processing. Other applications include pattern formation on animal coats, conservative shallow water models on the sphere and large-scale atmospheric flows over topography.

2.2.2 An Example

Let's suppose we are give some surface S. Without explicitly considering its geometries, we can define a PDE on the surface over time in terms of *intrinsic* in-surface differential operators [13]. For example, we might consider

$$u_t = f(u, \nabla_S u, \Delta_S u), \qquad (1.01)$$

where ∇_S is the intrinsic gradient. The Δ_S operator is the $Lapalace - Beltrami$ operator, the intrinsic operator defined in the same fashion as the Laplacian. We are interested in propagating the solution of the PDE (1.01) on the surface over time. Therefore, we need to have a surface representation.

2.2.3 The Closest Point Representation of Surfaces and Curves

Definition:(Closest point function)
The closest point function is defined locally for any smooth surface. For example, given a 3D surface S, the closest point function $cp(x) : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ takes a point x and returns a point $cp(x) \in S$, which is closest in Euclidean distance to x. That is, $cp(x) = argmin_{\hat{x}\in S}||x - \hat{x}||_2$. If x is within a small range of a smooth surface, the closest point is usually unique (e.g., if we compute on a small band defined near a smooth surface); however, sometimes multiple closest points may occur. In that case, for example, the center of a circle or any point on the diagonal of a rectangle, the closest point can be defined as any one of the arbitrary points among all the closest points. Using a closest point representation allows us to treat open surfaces, surfaces of arbitrary codimention and have an explicit mapping from the embedding space to the surface.

The left diagram in Figure 2.1 shows an example of the closest point function for a curve S. A few points in the embedding space and corresponding closest points $cp(x)$ are indicated by the marks. The middle diagram in 2.1 [16] is an example of the closest point function for a circle of radius R. Obviously, all the points on the circle have the same distance to the origin O, so $cp(O)$ can be chosen arbitrarily to be any point on the curve. The right diagram in 2.1 is an example of the closest point function for a rectangle. Non-smooth curves (surfaces in 3D) can also be treated using the Closest Point Method; however, an analysis of the corresponding flow has yet to be carried out.

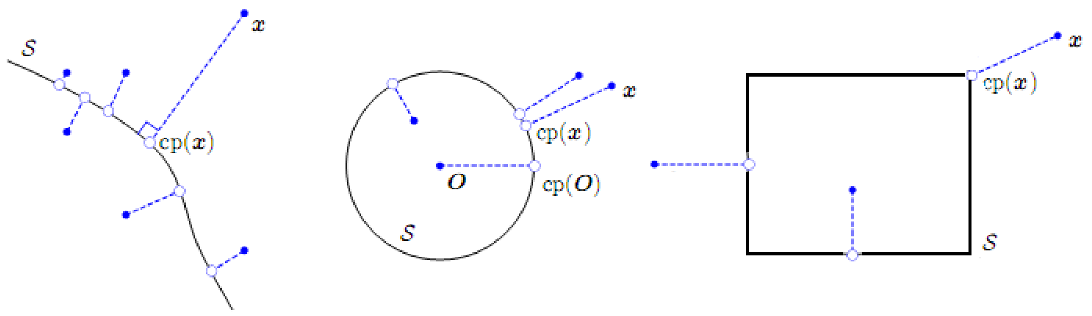

Figure 2.1: An illustration of several closest point function on some 2D curves [16],used with permission.

2.2.4 The Closest Point Extension

Suppose we have a function u defined on a 3D surface S. Then we can extend u to a function $\hat{u}$ defined on the whole 3D space by the following definition:

Definition:(Closest point extension)

Suppose S is a surface embedded in $\mathbb{R}^3$ and let $\hat{u} : S \rightarrow \mathbb{R}$ be a scalar function defined over S. Then the closest point extension of $\hat{u}$ is given by $u(x) = \hat{u}(cp(x))$.

Closest point extensions result in functions which are constant in the direction normal to the surface, at least within a neighborhood of a smooth surface. This fact leads to simplified derivative calculations in the embedding space [21].

2.2.5 Relationship between intrinsic differential operators and their Cartesian counterparts

Since the PDE is defined on a surface, operators such as the gradient and the divergence are naturally intrinsic to the surface. Therefore, they depend on the surface itself. However, the closest point function gives a simple way of relating these surface derivative operators to their standard Cartesian counterparts.

Gradient operators

Let ∇ denote the standard gradient in $\mathbb{R}^3$ and let ∇_S denote the gradient intrinsic to the surface S. For a point x on the surface, we have $\nabla u(x) = \nabla_S u(cp(x))$. This happens because the function $u(cp(x))$ is constant in the normal direction and therefore only varies along the surface, that is, at any point x on S, intrinsic surface gradients $\nabla_S u(x)$ are the same as gradients of $u(cp(x))$.

Divergence operators

Let $\nabla_s \cdot$ denote the divergence operator intrinsic to the surface S and let v be any vector field on $\mathbb{R}^3$ that is tangent to S and also tangent at all surfaces displaced by a fixed distance from S (all surfaces defined as level sets of the distance function to S). Then for any point x on S we have $\nabla \cdot v(x) = \nabla_S \cdot v(x)$. For the details of the proof, refer to [21].

2.3 The Closest Point Method

The equivalance of gradients, divergence and other differential operators now can be utilized to deal with surface differential operators by evaluating the corresponding differential operator in the embedding space $\mathbb{R}^3$. With all the operators defined above, we are now in a position to define the Closest Point Method. As an illustration, suppose we have a surface PDE taking the form

$$u_t(x,t) = f\left(x, t, u(x,t), \nabla_S u(x,t), \nabla_S \cdot \left(\frac{\nabla_S u(x,t)}{|\nabla_S u(x,t)|}\right)\right), \qquad (1.21a)$$

$$u_t(x,0) = u_0(x) \qquad (1.21b)$$

for $x \in S$ and $t \geq 0$. This is a general PDE which involves second-order flows. According the principles described in the last section, we are able to replace the surface gradients and divergence operators by the standard Cartesian operators in the embedding 3D space which yields:.

$$u_t(x,t) = f\left(cp(x), t, u(cp(x),t), \nabla u(cp(x),t), \nabla \cdot \left(\frac{\nabla u(cp(x),t)}{|\nabla u(cp(x),t)|}\right)\right), \qquad (1.22a)$$

$$u(x,0) = u_0(cp(x)) \qquad (1.22b)$$

for $x \in \mathbb{R}^3$ and $t \geq 0$. The solution of the PDE (1.21) and the PDE (1.22) will be identical on the surface in the sense that if $u_1(x,t)$ is a solution of (1.21) for $x \in S$ and $u_2(x,t)$ is a solution of PDE (1.22) for $x \in \mathbb{R}^3$ then $u_1(x,t) = u_2(x,t)$ for $t \geq 0$ and points on the surface $x \in S$.

2.3.1 The Explicit Closest Point Method

Now the PDE (1.22) can be discretized directly. To do this, consider the following PDE:

$$u_t(x,t) = f\left(x, t, u(x,t), \nabla u(x,t), \nabla \cdot \left(\frac{\nabla u(x,t)}{|\nabla u(x,t)|}\right)\right), \qquad (1.31a)$$

$$u(x,0) = u_0(cp(x)). \qquad (1.31b)$$

At time $t = 0$, the right-hand-sides of the PDEs (1.22a) and (1.31a) will be equal for all $x \in \mathbb{R}^3$. Thus, we could advance (1.22a) by a single forward Euler time step of size Δt by instead advancing (1.31a). This idea can be used to construct an algorithm valid for any time t_n. To do this, we start from a closest point extension of the solution u^n at time t_n and then take one forward Euler step to advance to time $\hat{u}^{n+1}$. After this step, $\hat{u}^{n+1}$ will not be constant in a direction normal to the surface. Another closest point extension of $\hat{u}^{n+1}$ is therefore carried out, $u^{n+1}(x) = \hat{u}^{n+1}(cp(x))$, so that $\hat{u}^{n+1}(cp(x))$ remains constant in a direction normal to the surface. Notice that the same procedure can be repeated by alternating the closest point extensions and propagating the solution forward via forward Euler to any desired time.

Now we can summarize the semi-discrete (discrete in time, continuous in space) explicit Closest Point Method.

2.3.2 The Explicit Closest Point Method Algorithm

1. Perform a forward Euler time step

$$\hat{u}^{n+1} = u^n + \Delta t f\left(cp(x), t_n, u^n(x), \nabla u^n(x), \nabla \cdot \left(\frac{\nabla u^n}{|\nabla u^n|}\right)\right). \qquad (1.32a)$$

2. Perform a closest point extension for each point in $\mathbb{R}^3$

$$u^{n+1}(x) = \hat{u}^{n+1}(cp(x)). \qquad (1.32b)$$

To solve the PDE, we still need a spatial discretization. In general, various finite difference schemes can be used. We emphasize that Equation (1.32a) only gives a valid evolution for one explicit time step, so it is necessary to perform another closest point extension before performing the next iteration of the algorithm. This ensures that the solution is consistent with the exact solution on the surface.

2.4 Interpolations

Although a regular Cartesian grid x_{ijk} is used, the closest point $cp(x)$ is usually not a grid point. Therefore, an interpolation step must be carried out. Specifically, we need an interpolation step to approximate the value at any point (non-grid point) by the grid points in some neighborhood [8].

2.4.1 1D Interpolation

The most common interpolation is to fit $p+1$ grid points with a polynomial of degree p, where $p \in \mathbb{N}, p \geq 0$. The interpolant for grid points surrounding x is evaluated at x. For example, the simplest linear ($p = 1$) interpolant is a straight line connecting two neighboring grid points x_i and x_{i+1} with $x_i \leq x < x_{i+1}$. Assuming the underlying function u is sufficiently smooth, for a degree p polynomial, the error generated by interpolation will be $O(\Delta x^{p+1})$ where Δx is the spacing between grid points. Degree p interpolation requires $p+1$ grid points around x, and these points are chosen as close as possible to x to minimize the $Runge\ phenomenon$ of oscillations at the far ends of the interpolant. There are many different kinds of techniques that can be used for interpolation, such as the Newton divided difference Method. One of the most efficient methods is the $Barycentric\ Lagrange\ method$[2].

Figure 2.2 is an illustration of 1D interpolation. The value of a function u is given at five grid points. To approximate the function at the interpolation point, a linear and a fourth-order polynomial are used.

Figure 2.3 shows the choice of the grid points used for interpolating polynomials from degree 0 to 4. The dots are the grid points and $\diamond$ is the point of interpolation. In all cases, the $p+1$ grid points closest to the interpolation point are chosen for degree p interpolation [16].

2.4.2 Barycentric Lagrange Interpolation

The $Barycentric\ Lagrange\ formula$ is a degree p interpolant through grid points $x_0, ..., x_p$ (equispaced in my case). Its evaluation at an interpolation point x is

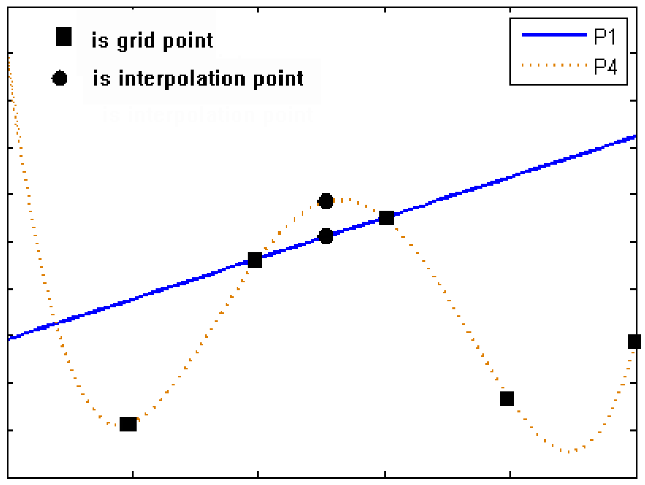

Figure 2.2: The figure shows 1D linear (solid blue) and 4th-degree interpolation (dotted red).

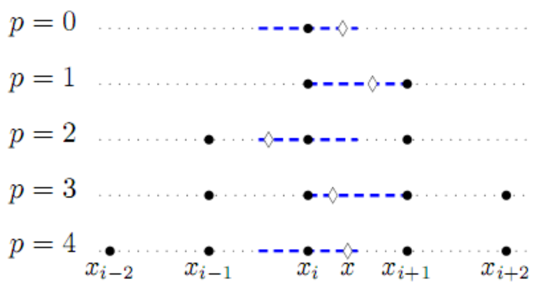

Figure 2.3: The pattern of grid points selected for degree p interpolation is illustrated here [16], used with permission.

$$u(x) \approx \frac{\sum_{j=0}^{p} \frac{w_j^{bc}}{x-x_j} u_j}{\sum_{j=0}^{p} \frac{w_j^{bc}}{x-x_j}}, \qquad w_j^{bc} = (-1)^j \binom{p}{j}$$

where the w_j^{bc} are the barycentric weights (which in this case are extremely simple due to the equispaced grid). Although $x - x_j$ appears in the denominators, the barycentric formula is always stable for x close to x_j. A straightforward special case needs to be picked out if x lies exactly on a grid point [2].

An attractive feature of the barycentric interpolant is that the coefficients of u_i are constants, independent of the values u_i to be interpolated. More precisely, the weights depend only on the grid points, x_i, the interpolation point, x, and the degree of the polynomial used for the interpolation. Because the Closest Point Method repeatedly interpolates with different data values, u_i, but with the interpolation points remaining fixed, this form is particularly efficient. All the weights $w_j = \frac{w_j^{bc}}{x-x_j} / \sum_{i=0}^{p} \frac{w_i^{bc}}{x-x_i}$ are pre-calculated allowing the interpolations to be performed as an inner product of the weights w_j with the grid point data u_i [2].

2.4.3 Higher Dimensional Interpolation

Higher dimensional interpolations are built in the same fashion as one-dimensional interpolations. For example, consider carrying out an interpolation on a 2D domain with cubic polynomials. First, the interpolation is carried out four times in the $x-$direction to obtain the corresponding values in a vertical line. Using these values, we can obtain the desired interpolated value using the standard one-dimensional interpolation [16].

Figure 2.4 shows the interpolation routine in 2D. Suppose we are interested in approximating the value at $\diamond$. First of all we use the grid points $\cdot$ to interpolate in the horizontal direction to get all 4 points $\triangle$. We use these $\triangle$ points to obtain the value at $\diamond$.

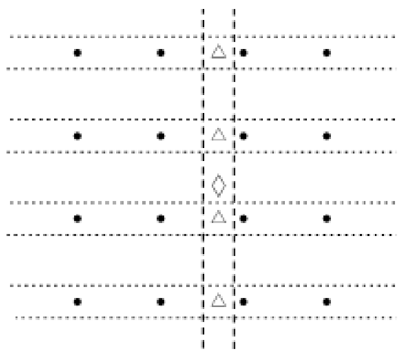

Figure 2.4: This figure shows that how a 2D interpolation works [16], used with permission. The four points($\triangle$) are interpolated by the dots in the same rows. Then the point at $\diamond$ is interpolated by the points located at the $\triangle's$

2.4.4 Banding

For the Closest Point Method, there is no need to calculate the evolution step or extension step over the entire embedding space. The calculation of evolution and extension step will be much more efficient if we use as few grid points as possible. Although there is absolutely no difference in the numerical result, a more efficient approach is to perform extension and evolution on a small narrow band around the surface. The band width depends on both the stencil width and the degree of the interpolation and is chosen so that all nodal values within the interpolation stencil have been accurately evolved [16]. Since the extension step will wipe out the values from last time step, all the grid points outside of the band will have no effect at all. Figure 2.5 shows the minimum bandwidths in 2D for a Closest Point Method computation for a surface in the vicinity of point $\diamond$ [16].
Assuming a standard five-point Laplacian is used in d-dimension and a polynomial of degree p is used for the interpolation, the general bandwidth λ is given by [21]

$$\lambda = \sqrt{(d-1)\left(\frac{p+1}{2}\right)^2 + \left(1 + \frac{p+1}{2}^2\right)}\Delta x.$$

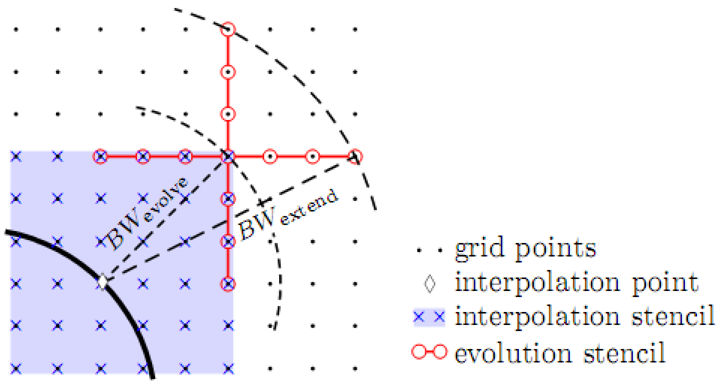

Figure 2.5: A basic standard bandwidth calculation indicating that the bandwidth depends on both the interpolation stencil and the evolution stencil [16], used with permission.

2.5 Chapter Summary

In this chapter, we gave a brief introduction to the Closest Point Method. This method will be an essential tool for solving the PDEs on surfaces appearing in Chapter 4.

Chapter 3

Segmentation on Images

3.1 Introduction

In this chapter, we review a recent technique for the segmentation of 2D images. See Tony F. Chan and Luminita A. Vese's paper, $Active\ Contours\ Without\ Edges$, for details.

The Chan-Vese model is based on techniques of curve evolution via level sets on the Mumford-Shah functional. The goal is to detect objects in images. The boundaries of the objects are not necessarily defined by the gradient. The algorithm evolves a level set function to minimize a certain energy rather than using the classical snake model [4] [5] [11]. The primary difference between this method and the classic active contour model is that the stopping condition does not depend on the gradient of the image but is rather related to a particular segmentation of the image. The numerical algorithm uses finite difference schemes, the initial condition can be arbitrary and interior contours can also be detected automatically.

3.2 Background of the model

To motivate the method, we consider a simple binary image u_0 composed of two distinct regions with piecewise-constant intensities. The intensities of the two regions are denoted by u_c^i and u_c^j. Let the region with intensity u_c^i represent the objects in the image. The boundary is denoted by C_0. Then, we have $u_0 = u_c^i$ inside C_0 and $u_0 = u_c^j$ outside C_0. We call the image domain Ω and the approximation of the boundary of the objects by C. Further, we set ω = inside(C) to be the region interior to the curve, and $\bar{\omega}$ = outside(C) to be the region exterior to the curve. The goal of segmenting the image will be achieved by minimizing an energy. Now consider the following functional [1]

$$\begin{aligned} F_1(C) + F_2(C) = & \int_{inside(C)} |u_0(x,y) - c_1|^2 dxdy \\ & + \int_{outside(C)} |u_0(x,y) - c_2|^2 dxdy, \end{aligned}$$

where C is any variable curve. The scalars c_1 and c_2 are the constants depending on C which are the averages of u_0 inside curve C and outside of C respectively. In this simple case, if C is the boundary of the object C_0, obviously we have

$$F_1(C_0) + F_2(C_0) = 0 = inf_C\{F_1(C) + F_2(C)\}.$$

Therefore, if we can find some curve C which minimizes the functional $F_1(C) + F_2(C)$, then we have found the desired objects in the image.

Figure 3.1 illustrates all possible positions of the curve.

a.)If curve C is outside the object, then we have $F_1(C) > 0, F_2(C) = 0.$
b.)If curve C is inside the object, then we have $F_1(C) = 0, F_2(C) > 0.$
c.)If curve C is partly inside and partly outside the object, then we have $F_1(C) > 0, F_2(C) > 0.$
d.)If curve C is the boundary of the object, then we have $F_1(C) = 0, F_2(C) = 0.$

We see that d.), which corresponds to the case where the curve C is on the boundary of the object, the functional $F_1(C) + F_2(C) = 0.$

To make the algorithm more robust, we can add some regularizing terms corresponding to the length of the curve C. This leads us to the energy functional $F(c_1, c_2, C)$ defined by

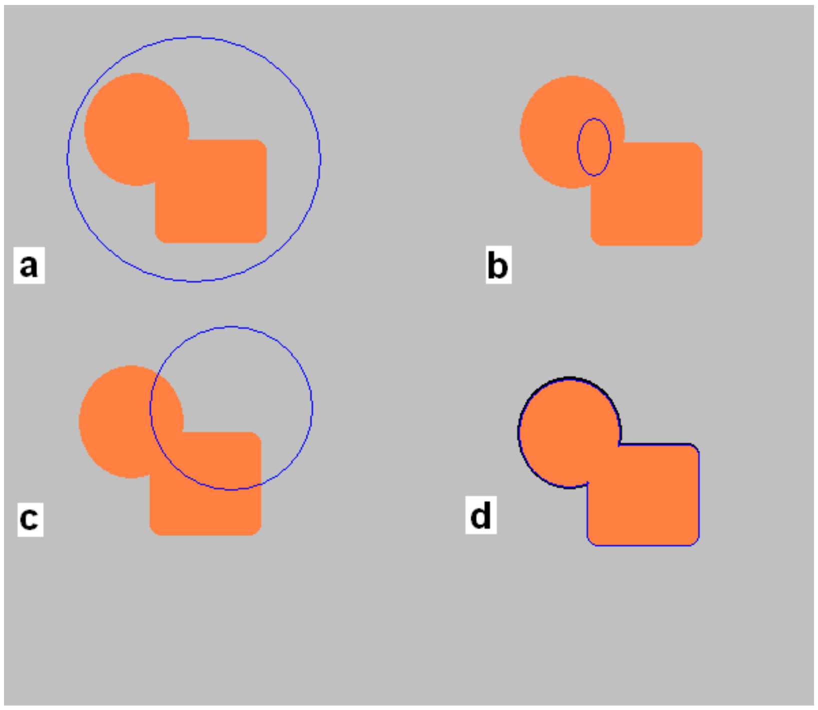

Figure 3.1: All possible configurations of the curve are illustrated here. Each curve separates the image into two regions.

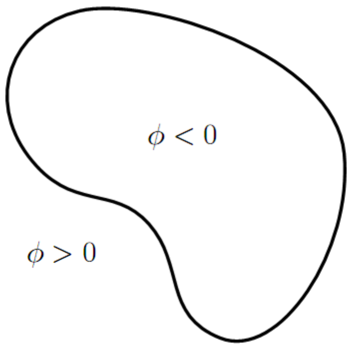

Figure 3.2: Level set function representation of a region on a 2D domain. The function ϕ is negative inside the curve and positive outside the curve [16], used with permission.

$$\begin{aligned} F(c_1, c_2, C) = \mu \cdot (\text{length of C}) \\ + \lambda_1 \int_{inside(C)} |u_0(x,y) - c_1|^2 dxdy \\ + \lambda_2 \int_{outside(C)} |u_0(x,y) - c_2|^2 dxdy, \end{aligned}$$

where $\mu \geq 0, \nu \geq 0, \lambda_1, \lambda_2 > 0$ are fixed parameters. In the numerical experiments, $\lambda_1 = \lambda_2 = 1$.

3.2.1 Mumford-Shah Functional

It is worth noting that the Mumford-Shah functional for segmentation is [18]

$$\begin{aligned} F^{MS}(u, C) = \mu \cdot (\text{length of C}) \\ + \lambda \int_{\Omega} |u_0(x,y) - u(x,y)|^2 dxdy \\ + \gamma \int_{\Omega \backslash C} |\nabla u(x,y)|^2 dxdy. \end{aligned}$$

As we can see, the Chan-Vese model with $\lambda_1 = \lambda_2 = \lambda$ is just a reduced form of the $Mumford-Shah$ formulation with the restriction that u is piecewise constant (this is also called the minimal partition problem) [6]. The Chan-Vese model looks for the best piecewise constant u fitting u_0, that is

$$u = \begin{cases} average(u_0) & \text{inside } C \\ average(u_0) & \text{outside } C \end{cases}$$

We now consider solving the Chan-Vese model using the level set method [19].

3.3 Level Set Formulation of the Model

The level set representation introduces a Lipschitz function $\phi : \Omega \to \mathbb{R}$ such that

$$\begin{aligned} C = \partial\omega &= \{(x,y) \in \Omega : \phi(x,y) = 0\}, \\ inside(C) = \omega &= \{(x,y) \in \Omega : \phi(x,y) > 0\}, \\ outside(C) = \bar{\omega} &= \{(x,y) \in \Omega : \phi(x,y) < 0\}. \end{aligned}$$

Having introduced ϕ, we may define u according to a variational model where the variable C is replaced by ϕ [24].

To accomplish this task, we introduce the Heaviside function H, and the one-dimensional Dirac measure δ_0. These are defined by

$$H(z) = \begin{cases} 1 & \text{if} \quad z \geq 0 \\ 0 & \text{if} \quad z < 0, \end{cases} \qquad \delta_0(z) = \frac{d}{dz}H(z).$$

We can now express the terms in the energy F as follows:

$$\text{Length}\{\phi = 0\} = \int_\Omega |\nabla H(\phi(x,y))| dxdy = \int_\Omega \delta_0(\phi(x,y))|\nabla\phi(x,y)| dxdy$$

and

$$\begin{aligned} \int_{\phi>0} |u_0(x,y) - c_1|^2 dxdy &= \int_\Omega |u_0(x,y) - c_1|^2 H(\phi(x,y)) dxdy, \\ \int_{\phi<0} |u_0(x,y) - c_2|^2 dxdy &= \int_\Omega |u_0(x,y) - c_2|^2 (1 - H(\phi(x,y))) dxdy. \end{aligned}$$

Combining these terms we find that the energy $F(c_1, c_2, \phi)$ can be written as

$$\begin{aligned} F(c_1, c_2, \phi) = \mu &\int_\Omega \delta_0(\phi(x,y))|\nabla\phi(x,y)|dxdy \\ + \lambda_1 &\int_\Omega |u_0(x,y) - c_1|^2 H(\phi(x,y))dxdy \\ + \lambda_2 &\int_\Omega |u_0(x,y) - c_2|^2 (1 - H(\phi(x,y)))dxdy. \end{aligned}$$

If we keep ϕ fixed and minimize the energy $F(c_1, c_2, \phi)$ with respect to the constants c_1 and c_2, the constants can be written as

$$c_1(\phi) = \frac{\int_\Omega u_0(x,y)H(\phi(x,y))dxdy}{\int_\Omega H(\phi(x,y))dxdy},$$

$$c_2(\phi) = \frac{\int_\Omega u_0(x,y)\left(1 - H(\phi(x,y))\right)dxdy}{\int_\Omega (1 - H(\phi(x,y)))dxdy}.$$

Assuming the curve has both interior and exterior area, then c_1 and c_2 are in fact given by [6]

$$\begin{aligned} c_1(\phi) &= \text{average}(u_0) \qquad \text{over} \qquad \{\phi \geq 0\}, \\ c_2(\phi) &= \text{average}(u_0) \qquad \text{over} \qquad \{\phi < 0\}. \end{aligned}$$

3.3.1 Existence of the Minimizer

The Chan-Vese model is one particular case of the minimal partition problem for which the existence of the solution has been proven [18]. We further note that the existence of the solution for the general Mumford-Shah segmentation problem has also been proved [17].

3.4 Variational Formulation

Now we can introduce a regularized versions of the functional H and δ_0, denoted as H_ϵ and δ_ϵ to compute the associated Euler-Lagrange equation for the unknown function ϕ. The associated regularized functional F_ϵ becomes

$$\begin{aligned} F_\epsilon(c_1, c_2, \phi) = \mu & \int_\Omega \delta_\epsilon(\phi(x,y))|\nabla\phi(x,y)|dxdy \\ & + \lambda_1 \int_\Omega |u_0(x,y) - c_1|^2 H_\epsilon(\phi(x,y))dxdy \\ & + \lambda_2 \int_\Omega |u_0(x,y) - c_2|^2 (1 - H_\epsilon(\phi(x,y)))dxdy. \end{aligned}$$

Keeping c_1 and c_2 fixed, and minimizing F_ϵ with respect to ϕ, we get the associated Euler-Lagrange equation for ϕ. Introducing an artificial time $t \geq 0$ and an initial contour $\phi_0(x,y)$. We obtain the corresponding gradient descent equations:

$$\begin{aligned} \frac{\partial\phi}{\partial t} = \delta_\epsilon(\phi)\Bigg[& \mu\,\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right) - \lambda_1(u_0(x,y) - c_1)^2 \\ & + \lambda_2(u_0(x,y) - c_2)^2\Bigg] \text{ in } (0,\infty)\times\Omega, \\ \phi(0,x,y) = \phi_0(x,y) & \text{ in } \Omega, \qquad \frac{\partial\phi}{\partial n} = 0 \text{ on } \partial\Omega. \end{aligned} \tag{3.1}$$

There are many choices of regularization functions. In this thesis, the experiments are carried out with this particular regularization of H and its Dirac delta function [6]

$$\begin{aligned} H_\epsilon &= \frac{1}{2}\left(1 + \frac{2}{\pi} arctan(\frac{z}{\epsilon})\right), \\ \delta_\epsilon &= \frac{dH}{dz} = \frac{\epsilon}{\pi(\epsilon^2 + z^2)}. \end{aligned}$$

This particular regularization function is chosen because this H has a global support and tends to compute a global minimizer [6]. If a regularization with a small support is chosen, then the Euler-Lagrange equation for ϕ acts only locally on a few level curves around its initial curve. This tends to give solutions which depend on the initial curve.

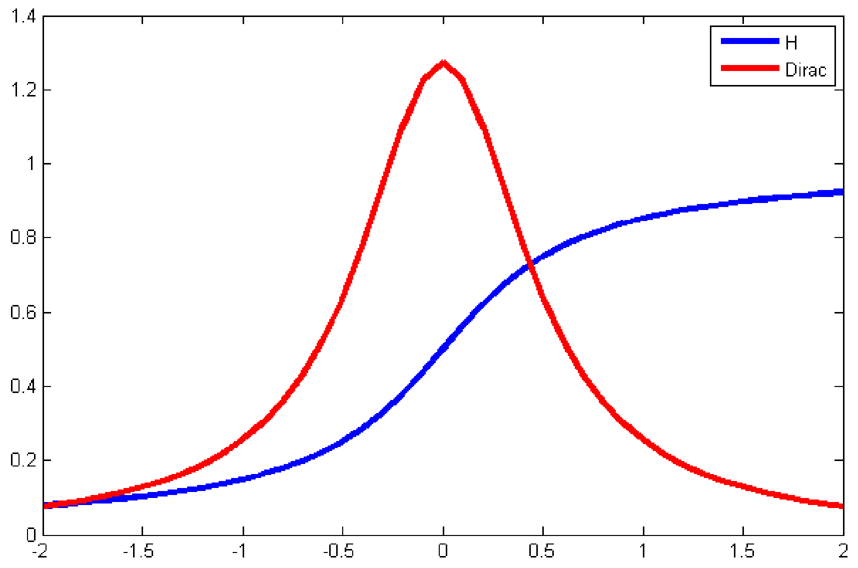

Figure 3.3: A plot of the chosen regularized Heaviside function H (blue) and its Dirac delta function (red).

3.4.1 Numerical Discretization

A finite difference implicit scheme (Backward Euler) is used to discretize the equation. Let h be the grid spacing and Δt be the time step size. We have

$$\begin{aligned} \frac{\phi_{ij}^{n+1} - \phi_{ij}^{n}}{\Delta t} = \delta_h(\phi_{ij}^{n}) \Bigg[\\ \frac{\mu}{h^2} \Delta_-^x \frac{\Delta_+^x \phi_{ij}^{n+1}}{\sqrt{\frac{(\Delta_+^x \phi_{ij}^n)^2}{h^2} + \frac{(\Delta_c^y \phi_{ij}^n)^2}{4h^2}}} \\ + \frac{\mu}{h^2} \Delta_-^y \frac{\Delta_+^y \phi_{ij}^{n+1}}{\sqrt{\frac{(\Delta_+^y \phi_{ij}^n)^2}{h^2} + \frac{(\Delta_c^x \phi_{ij}^n)^2}{4h^2}}} \\ -\lambda_1 \big(u_{0,ij} - u_1(\phi^n)\big)^2 + \lambda_2 \left(u_{0,ij} - u_2(\phi^n)\right)^2 \Bigg], \end{aligned} \tag{3.2}$$

where

$$\Delta_{-}^{x}\phi_{i,j} = \phi_{i,j} - \phi_{i-1,j}, \Delta_{+}^{x}\phi_{i,j} = \phi_{i+1,j} - \phi_{i,j}$$
$$\Delta_{-}^{y}\phi_{i,j} = \phi_{i,j} - \phi_{i,j-1}, \Delta_{+}^{y}\phi_{i,j} = \phi_{i,j+1} - \phi_{i,j}$$
$$\Delta_{c}^{y}\phi_{i,j} = \phi_{i,j+1} - \phi_{i,j-1}, \Delta_{c}^{x}\phi_{i,j} = \phi_{i+1,j} - \phi_{i-1,j}$$

This linear system can be solved iteratively [6].

3.4.2 Reinitialization

When working with level sets and Dirac delta functions, a reinitialization procedure is sometimes necessary to prevent the level set function from becoming too flat or too sharp. Therefore, ϕ might need to be reinitialized to the signed distance function. On the other hand, reinitialization also has side-effects of increasing computational cost and may prevent interior contours from growing. The evolution equation for reinitialization is given by

$$\begin{cases} \psi_{\tau} = sign(\phi(t))(1 - |\nabla\psi|) \\ \psi(0, \cdot) = \phi(t, \cdot) \end{cases}$$

In most of the experiments, no reinitialization step is needed.

3.5 Numerical Experiment on a 2D Image

A simple numerical experiment for the segmentation of a synthetic binary image (Figure 3.4) is presented in Figure 3.5. In this simple experiment, the green curve represents the zero contour of the level set function at various time steps. Clearly it shows that the zero contour of the function converges to the right segmentation of this binary image.

3.6 Chapter Summary

In this chapter, a simple segmentation model based on the paper $Active\ Contours\ Without\ Edges$ by Tony F.Chan and Luminita A. Vese is reviewed. The fundamental idea of this algorithm is to minimize an energy via variational and level set techniques. The approach was applied to 2D

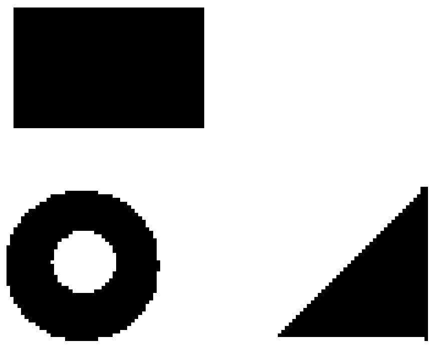
Figure 3.4: A simple binary image. The object is to use the Chan-Vese method to separate the black and the white regions.

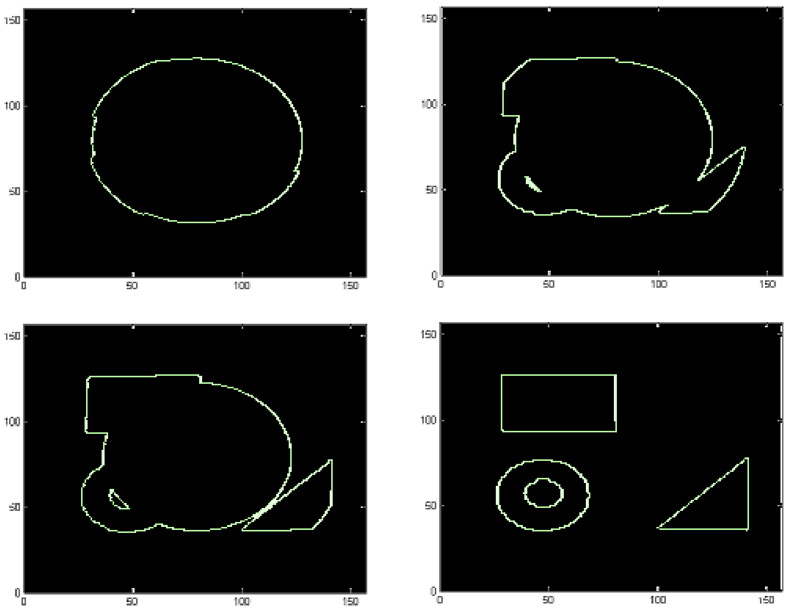

Figure 3.5: The zero contour of ϕ is dispayed at various times t.

images; however, this model is not just limited to 2D images. It can also be applied on 3D. With the Closest Point Method, an extension to images defined on surfaces is possible. This is the focus of the next chapter.

Chapter 4

Segmentation on a Surface

4.1 Introduction

With all the tools and background knowledge from the last two chapters, we are now able to move on to segmentation of objects and patterns on surfaces.

4.2 3D Segmentation Algorithm

As mentioned in the last chapter, the Chan-Vese active contours model is not just limited to 2D image segmentation. It can be applied to 3D images as well. The evolution of a curve in the 2D case becomes the evolution of a surface in 3D. There are some slight changes in the energy functional $F(s_1, s_2, \phi)$. It is now defined by

$$\begin{aligned} F(s_1, s_2, \phi) &= \mu \cdot (\text{Surface Area of } \phi) \\ &\quad + \lambda_1 \int_{inside(\phi)} |u_0(x, y, z) - s_1|^2 dxdydz \\ &\quad + \lambda_2 \int_{outside(\phi)} |u_0(x, y, z) - s_2|^2 dxdydz \end{aligned}$$

If the 3D object boundary is denoted by S, the Active Contour Model will be looking for the best approximation u of u_0 where

$$u = \begin{cases} average(u_0) & \text{inside } S \\ average(u_0) & \text{outside } S. \end{cases}$$

A similar variational formulation and discretization scheme can be applied to this 3D algorithm as was used in the 2D case.

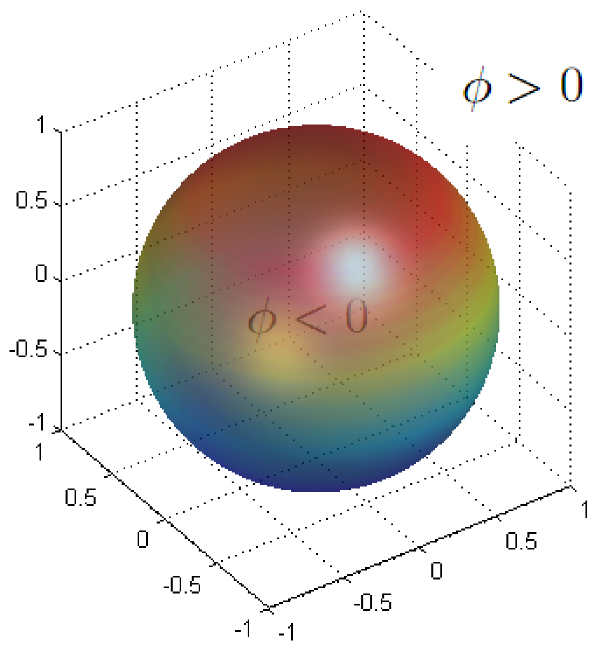

Figure 4.1: Level set representation of a spherical surface. The function ϕ is positive outside the ball and negative inside.

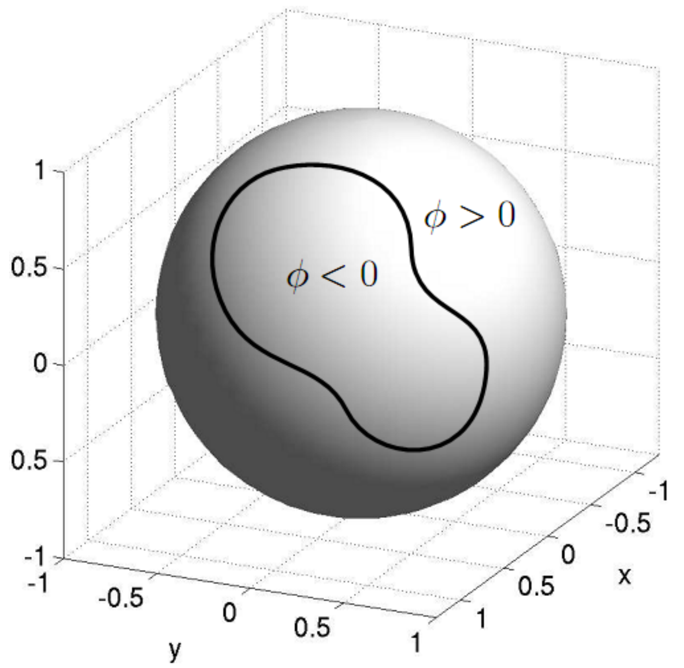

Figure 4.2: Zero contour of the level representation on a sphere separates the surface of the ball into two parts. The function ϕ is positive on one side of the area separated by the curve and negative on the other side [16], used with permission.

4.3 Chan-Vese Model on a Surface

Suppose some surface image data is given by a function $u_0 : S \to \mathbb{R}$. Assume the level set function used for the segmentation is $\phi : S \to \mathbb{R}$. Specifically, the zero contour of the function ϕ will divide the surface into two regions: $\{x \in S : \phi(x) > 0\}$ is the inside region of the zero contour and $\{x \in S : \phi(x) < 0\}$ is the outside region of the zero contour.

Figure 4.2 gives an illustration. The zero contour of the function ϕ separates the surface of a ball into two regions.

4.3.1 Formulation of the Energy Functional

According to the Chan-Vese model, the energy functional is

$$\begin{aligned} F(u_1, u_2, \phi) =& \mu \cdot (\text{length of } \phi\text{'s zero contour}) \\ &+ \lambda_1 \int_{\{x:\phi(x)>0\}} |u_0(x) - u_1|^2 dS \\ &+ \lambda_2 \int_{\{x:\phi(x)<0\}} |u_0(x) - u_2|^2 dS, \end{aligned}$$

where u_1 and u_2 are unknown scalars and $\mu \geq 0$, $\lambda_1, \lambda_2 > 0$ are fixed parameters. The energy can also be written as an integral over the entire surface by introducing the Heaviside function $H(z)$ and its Dirac measure $\delta(z)$. If we keep ϕ fixed and minimize the energy $F(u_1, u_2, \phi)$ with respect to the scalars u_1 and u_2,

$$\begin{aligned} u_1(\phi) &= \frac{\int_{\{x:\phi(x)>0\}} u_0(x) H(\phi(x)) dS}{\int_S H(\phi(x)) dS}, \\ u_2(\phi) &= \frac{\int_{\{x:\phi(x)<0\}} u_0(x) \left(1 - H(\phi(x))\right) dS}{\int_S (1 - H(\phi(x))) dS}. \end{aligned} \tag{4.1}$$

That is, u_1 and u_2 are the average values of u_0 over the inside and outside, respectively, of the zero contour of ϕ.

The regularized Heaviside function we introduce is the same as the one in Chapter 3

$$H_\epsilon(z) = \frac{1}{2}\left(1 + \frac{2}{\pi} \arctan\left(\frac{z}{\epsilon}\right)\right)$$

and the associated Dirac measure is once again

$$\delta_\epsilon = \frac{dH}{dz} = \frac{\epsilon}{\pi(\epsilon^2 + z^2)}.$$

Let $F_\epsilon(u_1, u_2, \phi)$ denote the resulting regularized functional. Keeping u_1 and u_2 fixed, and minimizing F_ϵ with respect to ϕ gives the corresponding Euler–Lagrange equation. These will be solved by evolving the gradient descent equations. The corresponding energy function $F(u_1, u_2, \phi)$ becomes

$$\begin{aligned} F_\epsilon(u_1, u_2, \phi) = \mu \int_S & \delta_\epsilon(\phi)|\nabla\phi| dS \\ &+ \lambda_1 \int_{\{x:\phi(x)>0\}} |u_0 - u_1|^2 H_\epsilon(\phi) dS \\ &+ \lambda_2 \int_{\{x:\phi(x)<0\}} |u_0 - u_2|^2 (1 - H_\epsilon(\phi)) dS. \end{aligned}$$

The corresponding gradient descent equations are

$$\begin{aligned} \frac{\partial\phi}{\partial t} = \delta_\epsilon(\phi)\Big[\mu\,\nabla_S \cdot \left(\frac{\nabla_S\phi}{|\nabla_S\phi|}\right) &- \lambda_1(u_0(x) - u_1)^2 \\ &+ \lambda_2(u_0(x) - u_2)^2\Big] \text{ in } (0,\infty)\times S, \\ \phi(0,x) = \phi_0(x) \text{ in } S, \qquad & \frac{\partial\phi}{\partial n} = 0 \text{ on } \partial S, \end{aligned} \tag{4.2}$$

where t is an artificial time. In order to obtain a zero contour which accurately segments out the objects, the equation needs to be solved to its steady state.

4.4 Transformation of the PDE

Although the surface PDE is well posed, it is very complicated to solve due to the surface derivative and divergence operators. From Chapter 2, we know that this PDE can be solved by the Closest Point Method. In Section 2.1.5 the equivalence for the gradient and divergence operator was reviewed. These principles indicate that

$$\nabla_S \cdot \left(\frac{\nabla_S\phi}{|\nabla_S\phi|}\right) = \nabla \cdot \left(\frac{\nabla\phi(cp)}{|\nabla\phi(cp)|}\right),$$

for all points x on a smooth surface S. Embedding the problem in $\mathbb{R}^3$, and using the above to replace the in-surface curvature term, we can obtain the *embedding PDE*

$$\begin{aligned} \frac{\partial\phi}{\partial t} = \delta_\epsilon(\phi(cp))\Big[\mu\,\nabla \cdot \left(\frac{\nabla\phi(cp)}{|\nabla\phi(cp)|}\right) &- \lambda_1(u_0(cp) - u_1)^2 \\ &+ \lambda_2(u_0(cp) - u_2)^2\Big] \text{ in } (0,\infty)\times\Omega. \end{aligned} \tag{4.3}$$

One thing worth noticing is the disappearance of the homogeneous Neumann boundary conditions. For the Closest Point Method, the homogeneous Neumann boundary conditions do not need to be explicitly imposed; they are automatically generated by the method. No artificial boundary conditions at the edge of the computational domain are introduced either, because the computational domain includes only a narrow band of the surface.

4.4.1 Discretization

To discretize the PDE, we begin by selecting a computational domain Ω_c of discrete grid points: it is not necessary to solve on the whole domain, but just on a certain narrow band enveloping the surface [21, 14]. Most of the numerical experiments are carried out with cubic interpolation. Hence, the bandwidth is given by

$$\lambda = \sqrt{2 \cdot 2^2 + 3^2}\Delta x.$$

For the detail of the calculation of the bandwidth, please refer to Chapter 2. For each grid point in Ω_c, we need to construct a closest point function $cp(x)$ for the surface S. This will represent the surface. For simple surfaces such as a sphere, torus or cube, the closest point function has an easy analytical form. For triangulated surfaces, the closest point function can be computed efficiently by the algorithm outlined in [14].

To solve the embedding PDE (4.3) we alternate the following two steps:

1. Extend the solution off the surface to the computational domain using the closest point function. That is, replace ϕ by $\phi(cp)$ for each node on the computational domain. This *closest point extension* is an interpolation step, and is carried out using barycentric Lagrange interpolation [2].

2. Compute the solution to the embedding PDE (4.3) using standard finite differences on the Cartesian mesh for one time step. This step is just a step of the usual Chan–Vese algorithm in 3D for region segmentation.

The second step of the algorithm is discretization by a finite difference scheme which is semi-implicit in time. Following [6], it is

$$\begin{aligned}\frac{\phi_{ijk}^{n+1}-\phi_{ijk}^{n}}{\Delta t} &= \delta_h(\phi_{ijk}^{n})\Bigg[\\ &\frac{\mu}{h^2}\Delta_-^x \frac{\Delta_+^x \phi_{ijk}^{n+1}}{\sqrt{\frac{(\Delta_+^x\phi_{ijk}^n)^2}{h^2}+\frac{(\Delta_c^y\phi_{ijk}^n)^2}{4h^2}+\frac{(\Delta_c^z\phi_{ijk}^n)^2}{4h^2}}} \\ &+\frac{\mu}{h^2}\Delta_-^y \frac{\Delta_+^y \phi_{ijk}^{n+1}}{\sqrt{\frac{(\Delta_+^y\phi_{ijk}^n)^2}{h^2}+\frac{(\Delta_c^x\phi_{ijk}^n)^2}{4h^2}+\frac{(\Delta_c^z\phi_{ijk}^n)^2}{4h^2}}} \\ &+\frac{\mu}{h^2}\Delta_-^z \frac{\Delta_+^z \phi_{ijk}^{n+1}}{\sqrt{\frac{(\Delta_+^z\phi_{ijk}^n)^2}{h^2}+\frac{(\Delta_c^x\phi_{ijk}^n)^2}{4h^2}+\frac{(\Delta_c^y\phi_{ijk}^n)^2}{4h^2}}} \\ &-\lambda_1\big(u_{0,ijk}-u_1(\phi^n)\big)^2+\lambda_2\left(u_{0,ijk}-u_2(\phi^n)\right)^2\Bigg],\end{aligned} \tag{4.4}$$

where

$$\begin{aligned}\Delta_-^x\phi_{i,j,k}=\phi_{i,j,k}-\phi_{i-1,j,k}, \Delta_+^x\phi_{i,j,k}&=\phi_{i+1,j,k}-\phi_{i,j,k}\\ \Delta_-^y\phi_{i,j,k}=\phi_{i,j,k}-\phi_{i,j-1,k}, \Delta_+^y\phi_{i,j,k}&=\phi_{i,j+1,k}-\phi_{i,j,k}\\ \Delta_-^z\phi_{i,j,k}=\phi_{i,j,k}-\phi_{i,j,k-1}, \Delta_+^z\phi_{i,j,k}&=\phi_{i,j,k+1}-\phi_{i,j,k}\\ \Delta_c^y\phi_{i,j,k}&=\phi_{i,j+1,k}-\phi_{i,j-1,k},\\ \Delta_c^x\phi_{i,j,k}&=\phi_{i+1,j,k}-\phi_{i-1,j,k},\\ \Delta_c^z\phi_{i,j,k}&=\phi_{i,j,k+1}-\phi_{i,j,k-1}\end{aligned}$$

and h is the grid spacing.

The linear system (4.5) is solved by the iterative Gauss–Seidel method [1, 6]. Because we are interested in the steady state solution to (4.3), it is not necessary to iterate to convergence at each time step but merely to descend towards the minimum energy. The stopping condition can either monitored by observation of the zero contour or the criteria defined by the tolerance of the relative difference between each step of ϕ.

4.4.2 Gauss–Seidel iterative method

In order to solve this discretized PDE by the Gauss–Seidel iterative method, we need to rewrite the PDE as

$$\begin{aligned}\phi_{ijk}^{n+1} = \phi_{ijk}^{n} + \Delta t \delta_h(\phi_{ijk}^{n}) \Bigg[\\ \frac{\mu}{h^2}\Delta_-^x \frac{\Delta_+^x \phi_{ijk}^{n+1}}{\sqrt{\frac{(\Delta_+^x \phi_{ijk}^n)^2}{h^2} + \frac{(\Delta_c^y \phi_{ijk}^n)^2}{4h^2} + \frac{(\Delta_c^z \phi_{ijk}^n)^2}{4h^2}}} \\ + \frac{\mu}{h^2}\Delta_-^y \frac{\Delta_+^y \phi_{ijk}^{n+1}}{\sqrt{\frac{(\Delta_+^y \phi_{ijk}^n)^2}{h^2} + \frac{(\Delta_c^x \phi_{ijk}^n)^2}{4h^2} + \frac{(\Delta_c^z \phi_{ijk}^n)^2}{4h^2}}} \\ + \frac{\mu}{h^2}\Delta_-^z \frac{\Delta_+^z \phi_{ijk}^{n+1}}{\sqrt{\frac{(\Delta_+^z \phi_{ijk}^n)^2}{h^2} + \frac{(\Delta_c^x \phi_{ijk}^n)^2}{4h^2} + \frac{(\Delta_c^y \phi_{ijk}^n)^2}{4h^2}}} \\ -\lambda_1 \big(u_{0,ijk} - u_1(\phi^n)\big)^2 + \lambda_2 \left(u_{0,ijk} - u_2(\phi^n)\right)^2 \Bigg],\end{aligned} \tag{4.5}$$

$$\begin{aligned}\phi_{i,j,k}^{n+1} = \phi_{i,j,k}^{n} + \Delta t \Bigg\{ \frac{\mu}{h^2}\Big[C_1(\phi_{i+1,j,k}^n - \phi_{i,j,k}^n) + C_2(\phi_{i-1,j,k}^n - \phi_{i,j,k}^n) + \\ C_3(\phi_{i,j+1,k}^n - \phi_{i,j,k}^n) + C_4(\phi_{i,j-1,k}^n - \phi_{i,j,k}^n) + C_5(\phi_{i,j,k+1}^n - \phi_{i,j,k}^n) + \\ C_6(\phi_{i,j,k-1}^n - \phi_{i,j,k}^n)\Big] - \lambda_1(u_{0,i,j,k} - u_1)^2 + \lambda_2(u_{0,i,j,k} - u_2)^2 \Bigg\}\end{aligned}$$

where $C_1, C_2, C_3, C_4, C_5, C_6$ are the coefficients for each stencil point. Let

$$C_7 = 1 - \frac{\Delta t \mu \delta_h(\phi)}{h^2}(C_1 + C_2 + C_3 + C_4 + C_5 + C_6).$$

Now the PDE can be rewritten as:

$$\begin{aligned}\phi_{i,j,k}^{n+1} = C_7\phi_{i,j,k}^n + \Delta t \Bigg\{ \frac{\mu}{h^2}\Big[C_1\phi^n + C_2\phi_{i-1,j,k}^n + \\ C_3\phi_{i,j+1,k}^n + C_4\phi_{i,j-1,k}^n + C_5\phi_{i,j,k+1}^n + \\ C_6\phi_{i,j,k-1}^n\Big] - \lambda_1(u_{0,i,j,k} - u_1)^2 + \lambda_2(u_{0,i,j,k} - u_2)^2 \Bigg\}\end{aligned}$$

For each iteration of the Gauss–Seidel method, the $\phi_{i,j,k}$ node will be updated accordingly, and the newly updated node point will replace the old value at each step. For example, we update the point $\phi^{n+1}_{i+1,j+1,k+1}$ using $\phi^{n+1}_{i,j,k}$ instead of $\phi^{n}_{i,j,k}$. The detail of the convergence is covered in [1]. In the numerical experiments, only a few iterations are needed due to the fast convergence.

4.4.3 Efficiency Improvement

Our ultimate goal is to find the steady state of the gradient descent equations (4.2). To achieve this objective, it is sometimes helpful to break the computation into two phases. In the first phase, we are interested in a fast method which gives a qualitatively improved segmentation. In this phase, enhanced computational speed can be obtained by dropping the interpolation step in the algorithm. The second phase needs to be consistent with the gradient descent equations (4.2) and therefore must include the interpolation step. This gives a result which can be used as a starting condition for the second phase. We can save large amounts of the computational time by skipping the interpolation step, especially for some regular smooth surfaces. In most of the cases, it gives us promising results close to the final steady state solution; however, in order to get a more accurate and refined segmentation such as on some irregular triangulated surfaces, we need to include the interpolation step. This gives us a way to adjust the efficiency and accuracy of the segmentation. If we need fast coarse segmentation, we can skip the interpolation phase. If we need more accurate and detailed segmentation, we make use of the interpolation phase.

4.5 Chapter Summary

In this chapter, the algorithm used for segmentation on surfaces is presented. This model is based on the Chan-Vese active contour model combined with the Closest Point Method. The Closest Point Method gives a simple way to extend the 3D Chan-Vese model to the problem of segmenting out objects defined in textures on surfaces.

Chapter 5

Numerical Experiments

5.1 Introduction

In this chapter, different kinds of numerical experiments for segmentation on surfaces are presented. This includes segmentation on closed and open smooth surfaces such as spheres and hemispheres, surfaces with corners such as cubes and a particular triangulated surface.

5.2 Numerical Parameters

In the numerical experiments, we take $\lambda_1 = \lambda_2 = 1$ and $h = 1$, $\Delta t = 0.1$. The range of the surface data u_0 is $[0, 255]$. In all instances, the initial contour is given by the intersection of the surface with that of a cylinder of radius 15 units. The parameters are chosen suggest by the Chan-Vese model. The closest point extension steps are performed using barycentric Lagrange interpolation [2] with cubic polynomials. Computations are performed in Matlab and visualizations are done using either Matlab or Paraview [9].

5.3 Segmentation on Smooth Surfaces

5.3.1 Binary Image on Sphere

In this experiment, a synthetic binary pattern is created on a sphere. The sphere is 36 units in diameter with $\mu = 0.02 \cdot 255^2$. This is just a warm-up test, since the image is binary and the surface is smooth.

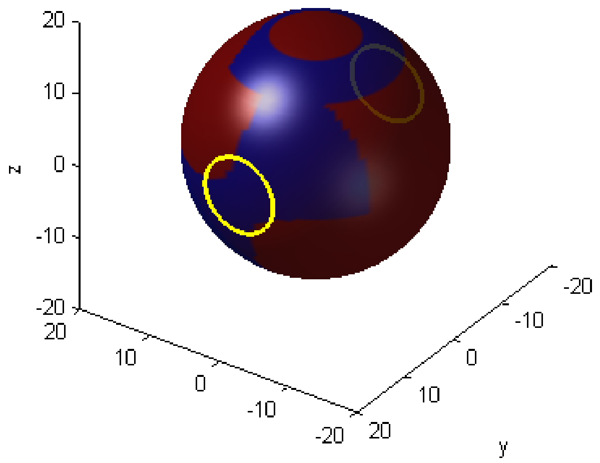

Figure 5.1: A sphere and the initial zero contour of the function.

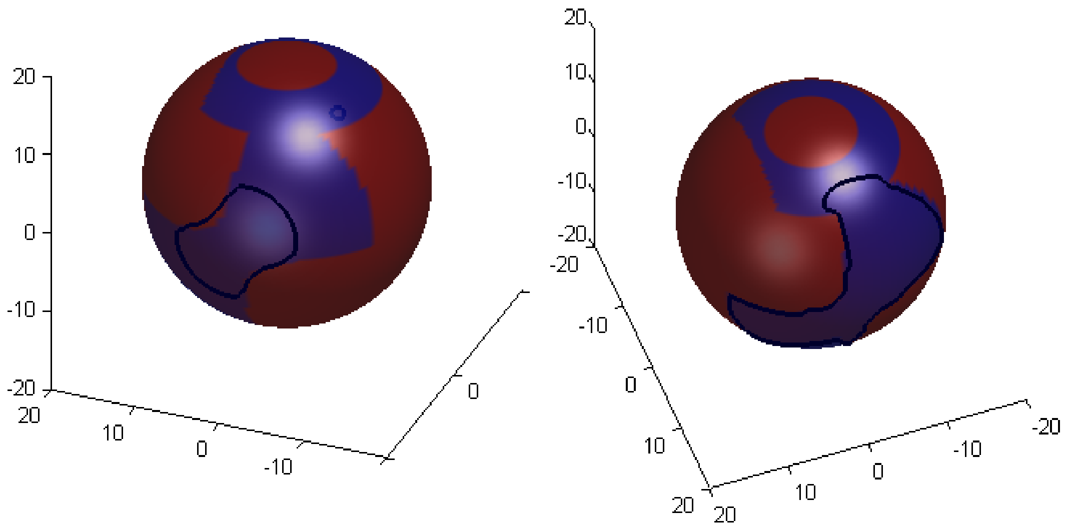

Figure 5.2: The evolving zero contour at times $t = 0.3, t = 0.6$.

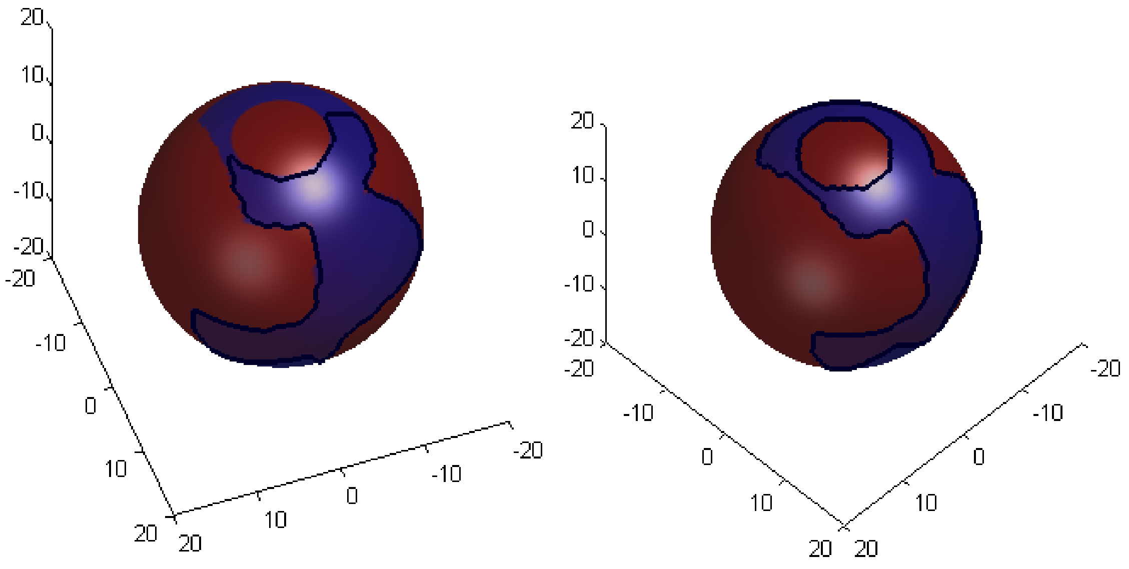

Figure 5.3: The evolving zero contour at times $t = 1, t = 1.2$.

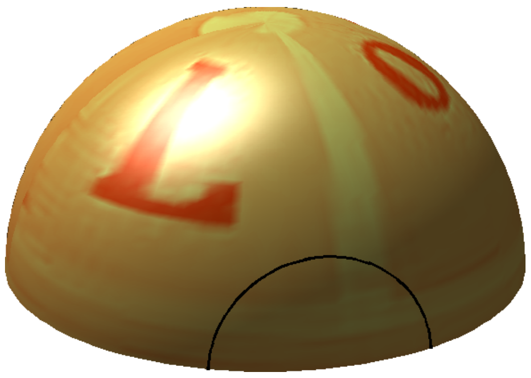

Figure 5.4: A hemisphere and the initial contour of the function.

5.3.2 Segmentation on an Open Smooth Surface

In this experiment, an open surface is selected and a noisy 2D image is mapped onto the hemisphere. The hemisphere is 72 units in diameter and $\mu = 0.04 \cdot 255^2$. The significance of this experiment is that the surface is open and that there are some artifacts in the mapped image. We note that the Closest Point Method experiences no problem in solving the PDE on the open surface. The Chan-Vese model also ignores the artifacts and picks out the desired objects. See Figure 5.5 and 5.6.

5.4 Segmentation on a Cube

In this experiment, we consider grey-scale data on the surface of a cube. The top of the cube consists of a bitmap image of a galaxy and the other faces contain geometric shapes. The cube is 80 units on each side and $\mu = 0.1 \cdot 255^2$.

Figures 5.7 to 5.9 show the evolution of the zero contour of ϕ at various times. With this choice of μ, the evolution segments out the core of the galaxy, as well as each of the shapes on the surface. Different μ-values can lead to different segmentations of the galaxy. It is noteworthy that the underlying surface is non-smooth. The analysis of the Closest Point Method on non-smooth surfaces is ongoing, but our results here suggest that the method is stable and achieves qualitatively correct results in the context of image segmentation.

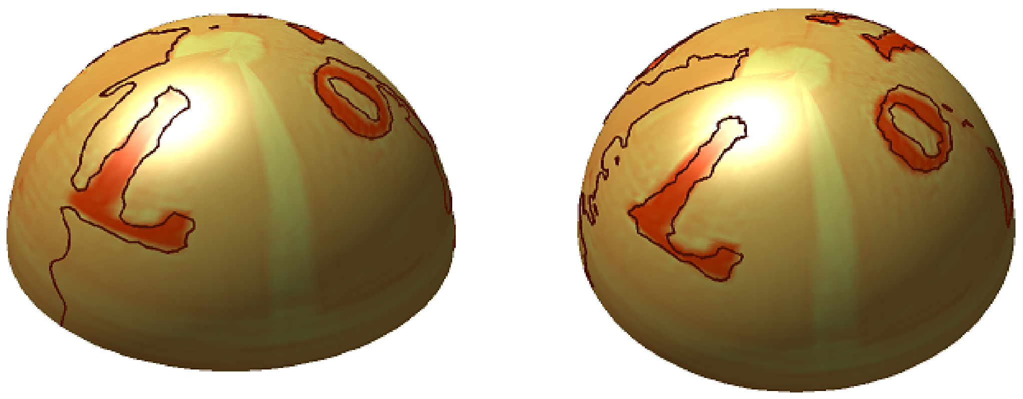

Figure 5.5: The evolving zero contour at times $t = 0.5, t = 1.1$.

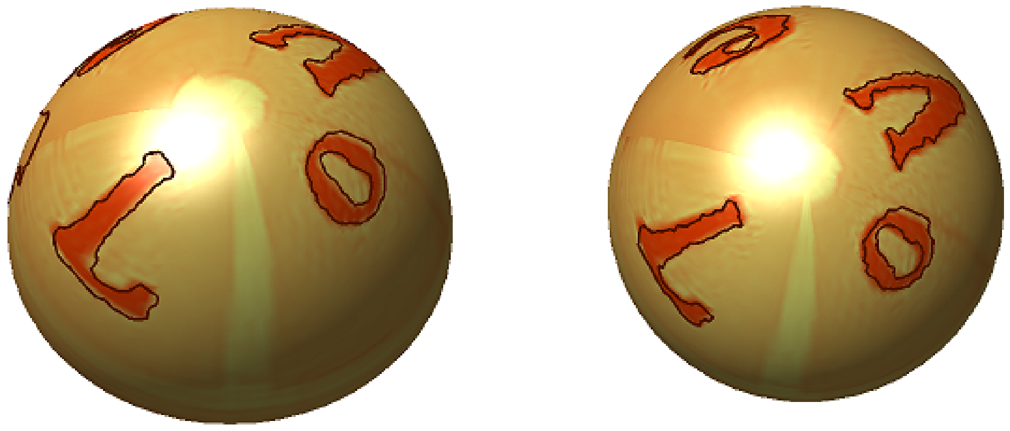

Figure 5.6: The evolving zero contour at times $t = 1.5, t = 2$.

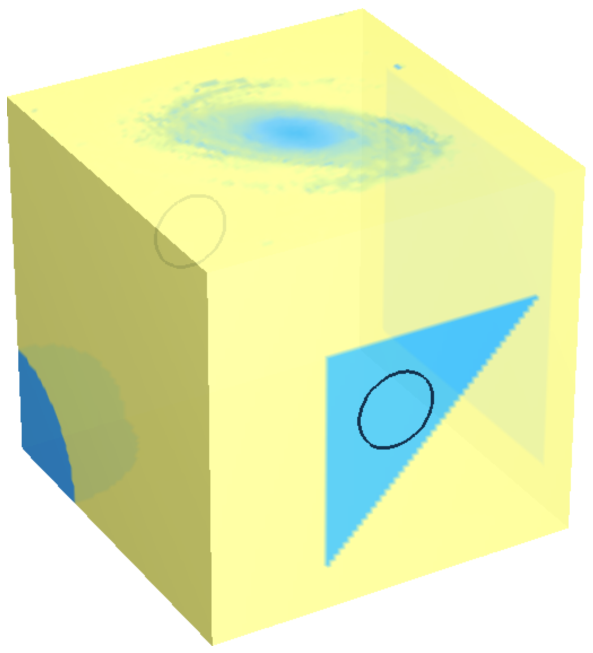

Figure 5.7: A cube and the initial zero contour of the function ϕ.

5.5 Segmentation on a Triangulated Surface

The algorithm can also be applied to complicated triangulated shapes, such as Annie Hui's pig [10] (here smoothed via Loop's algorithm within ParaView [9]). The surface data u_0 consists of various shapes that were manually generated.

5.5.1 Loop Algorithm

The original data on the pig is very coarse. In order to get better representation of the pig, we refine the pig by the Loop subdivision algorithm, which can be carried out by Para-View (See Figure 5.10). Figures 5.11 and 5.12 show the evolution of the zero level set of ϕ (shown as a white contour) on the pig at various times. The segmentation automatically finds the boundary between the two colored regions. It is worth emphasizing that the code for segmenting the triangulated pig is identical to that for our other examples: segmentation on a pig, cube, hemisphere or any other surface merely requires the input of the closest point function corresponding to the surface.

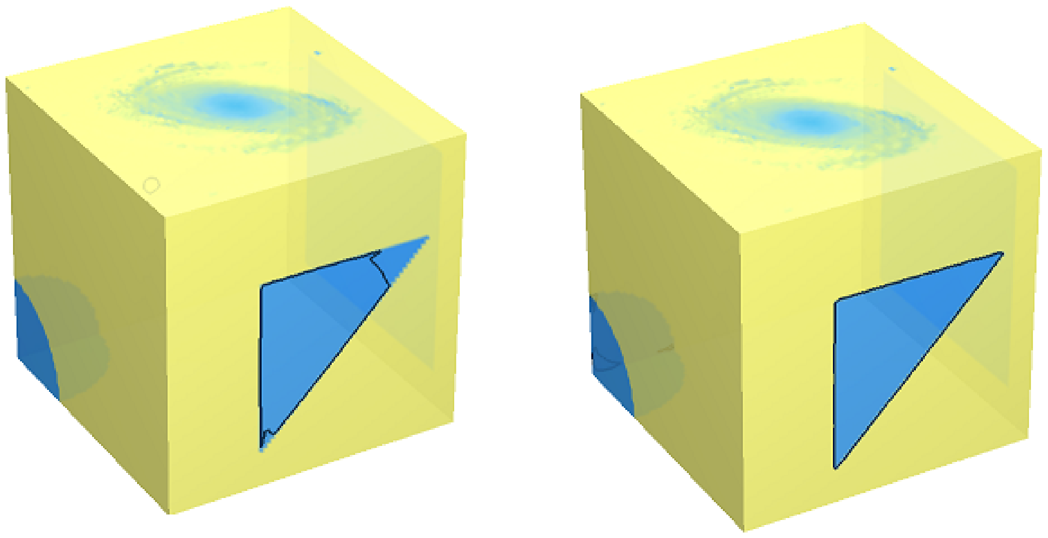
Figure 5.8: The evolving zero contour at times $t = 1.2, t = 1.8$.

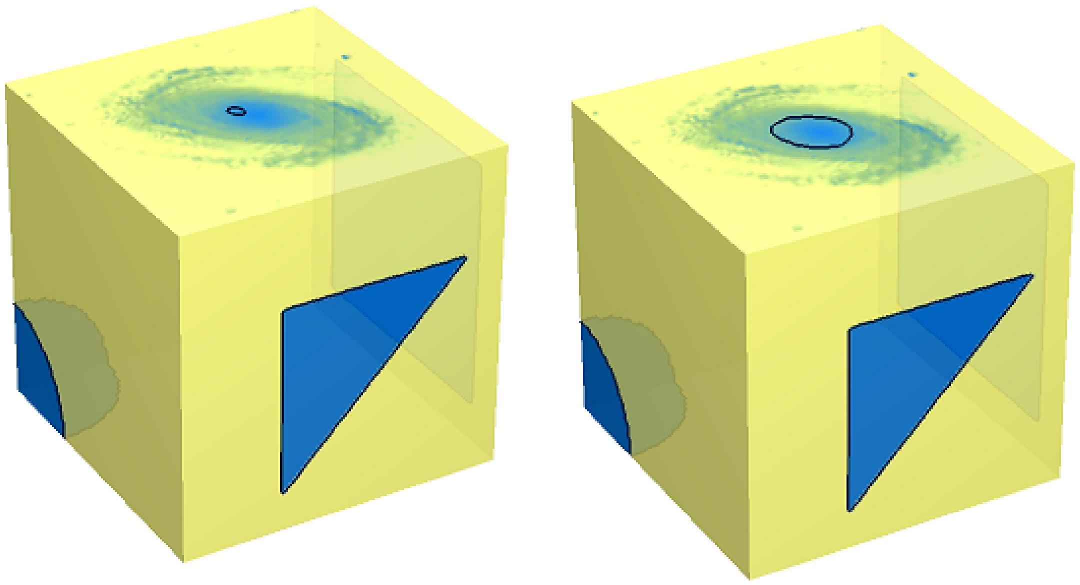

Figure 5.9: The evolving zero contour at times $t = 2.5, t = 4.8$.

5.6 Chapter Summary

Our numerical tests indicate that the effectiveness of applying the Closest Point Method to the Chan-Vese model. The results show that the PDE converges to the desired solution not only for simple surfaces such as a sphere and a cube, but it also works for more complicated triangulated surfaces.

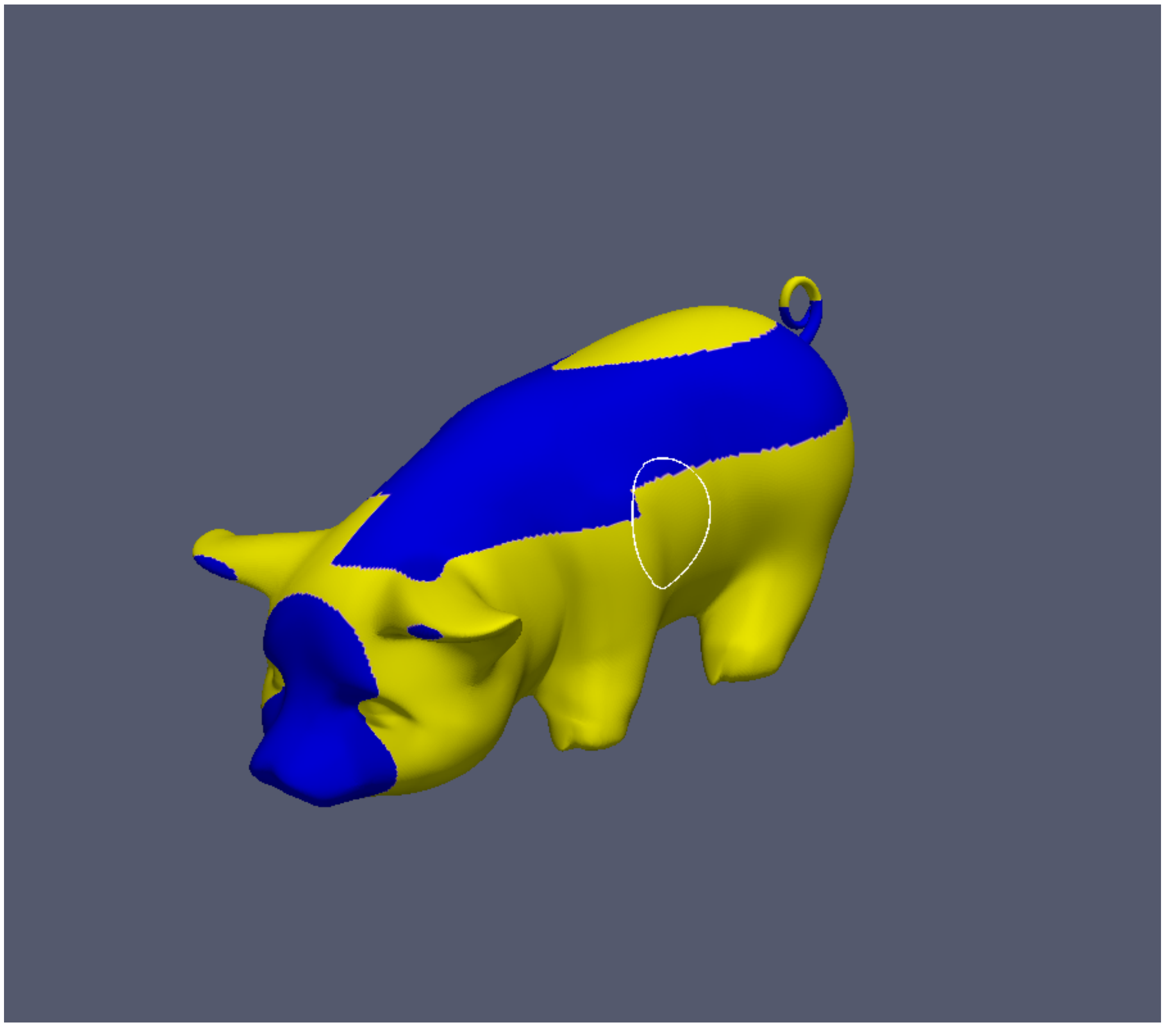

Figure 5.10: A triangulated surface of a pig and the initial zero contour of the function are shown.

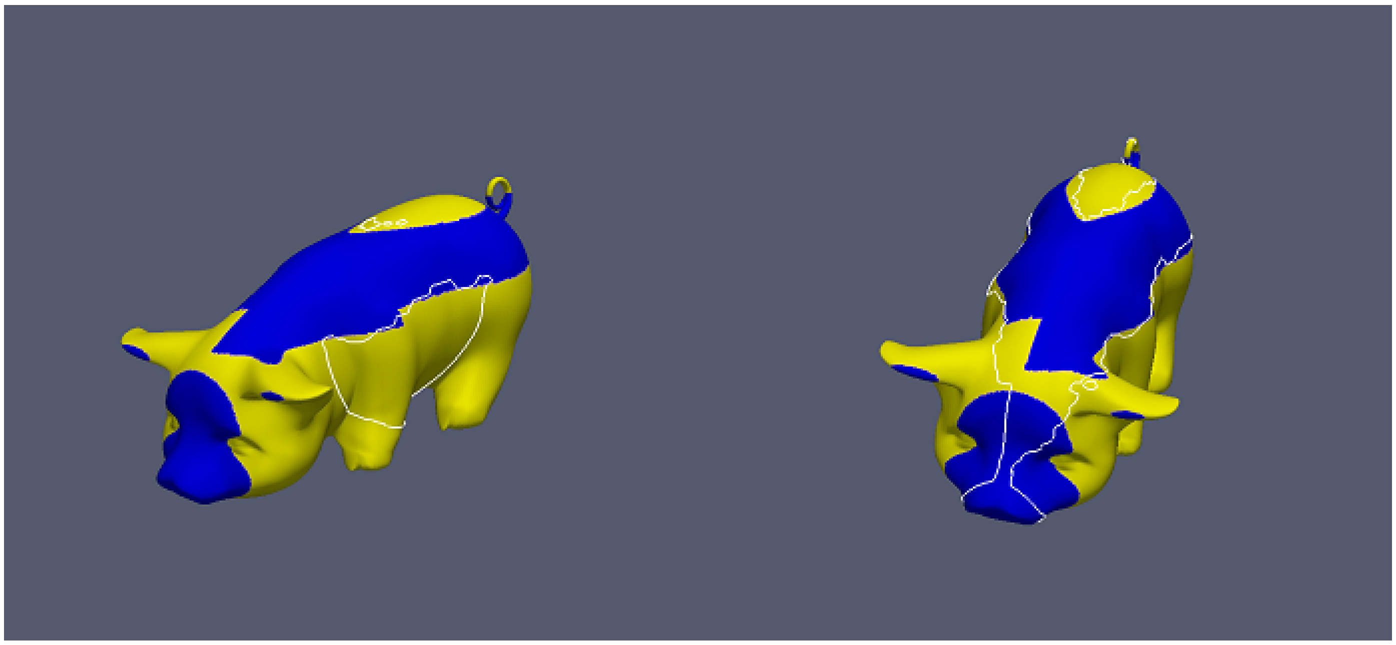
Figure 5.11: The evolving zero contour at times $t = 0.2, t = 2.8$.

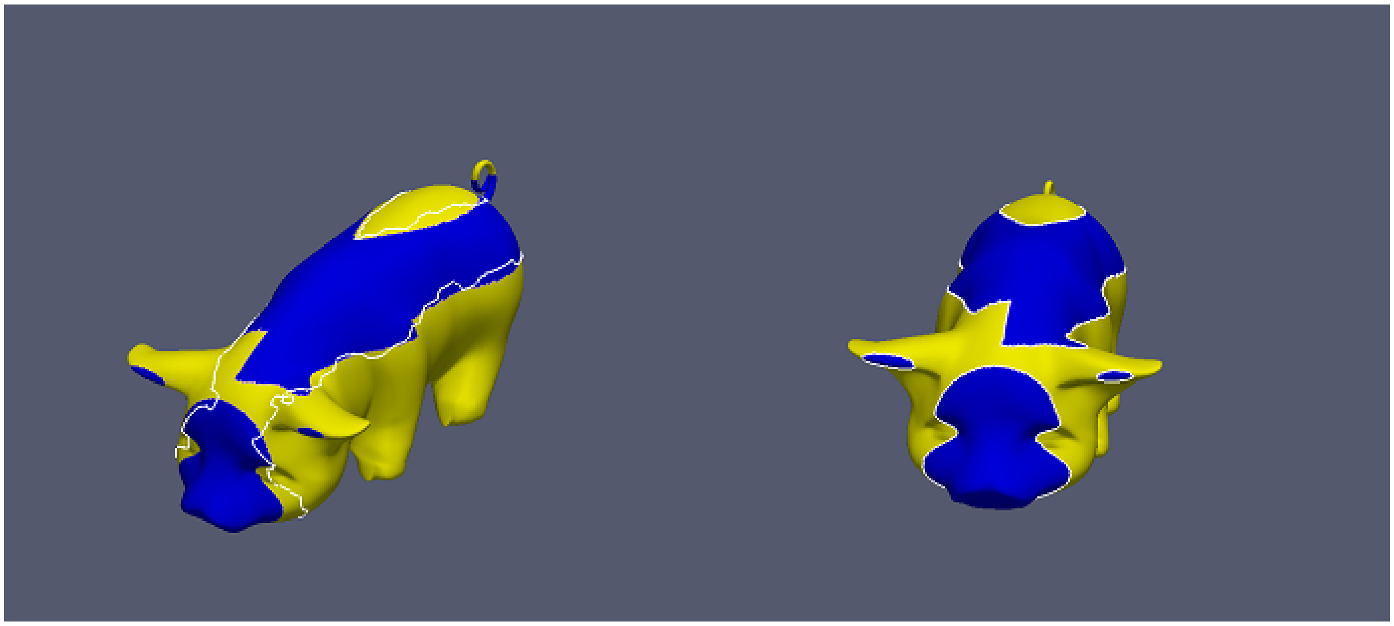
Figure 5.12: The evolving zero contour at times $t = 4.6, t = 35.8$.

Chapter 6

Conclusion

Recently, solving PDEs on surfaces has been gaining widespread interest and popularity. For example, surface PDEs arise in the segmentation of images on surfaces in 3D. In this thesis, the Chan-Vese Active Contour Model and the Closest Point Method are combined and applied to the segmentation on surfaces. A particularly simple and efficient way to solve surface PDEs is the Closest Point Method.

Chapter 2 illustrates how the Closest Point Method can be applied to solve PDEs on surfaces. Of particular relevance to this thesis is that the Closest Point Method provides a simple way to solve level set equations on surfaces using standard methods in 3D.

Chapter 3 gives a brief introduction of the Chan-Vese model of 2D image segmentation. It uses a variational model to minimize a simplified energy functional rather than the usual Mumford-Shah energy.

Chapter 4 combines the knowledge of Chapters 2 and 3. By using the Closest Point Method, the Chan-Vese model is now no longer limited to 2D or 3D image segmentation. It can now be applied to segment out images on general surfaces.

Finally, Chapter 5 contains a variety numerical experiments. It considers both open and closed smooth surfaces, a non-smooth surface and a triangulated surface. The method works well and all the final steady-state solutions clearly show the desired segmentation of the images.

A variety of topics remain to be explored. One of the most interesting questions in segmentation is how to utilize both image intensity and surface geometry to obtain improved segmentation on surfaces. For example, a term involving gradient of the surface could be added when formulating a new surface PDE to make use of the surface geometry.

Bibliography

[1] G. Aubert and L. Vese. A variational method in image recovery. *SIAM J. Numer. Anal.*, 34, 1997.

[2] J.-P. Berrut and L.N. Trefethen. Barycentric Lagrange interpolation. *SIAM Rev.*, 46(3), 2004.

[3] I. Bogdanova, X. Bresson, J.-P. Thiran, and P. Vandergheynst. Scale-space analysis and active contours for omnidirectional images. *IEEE Trans. Image Process.*, 16(7), 2007.

[4] V. Caselles, F. Catte, T. Coll, and F. Dibos. A geometric model for active contours in image processing. *Numer. Math*, 66, 1993.

[5] V. Caselles, R. Kimmel, and G. Sapiro. On geodesic active contours. *Int. J. Comput. Vis*, 22(1), 1997.

[6] T.F. Chan and L.A. Vese. Active contours without edges. *IEEE Trans. Image Process.*, 10(2), 2001.

[7] K. Hara, R. Kurazume, K. Inoue, and K. Urahama. Segmentation of images on polar coordinate meshes. In *Proc. ICIP07, International Conference on Image Processing*, 2007.

[8] Michael T. Heath. *Scientific Computing: An Introductory Survey*. McGraw-Hill, 1997.

[9] A. Henderson. *ParaView Guide, A Parallel Visualization Application*. Kitware, Inc., third edition, 2008.

[10] A. Hui. Annie Hui's pig in the AIM@SHAPE shape repository. `http://shapes.aimatshape.net`, 2008.

[11] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *Int. J. Comput. Vis*, 1, 1988.

[12] M. Krueger, P. Delmas, and G. Gimel'farb. Active contour based segmentation of 3D surfaces. In *Proc. European Conference on Computer Vision (ECCV)*, 2008.

[13] W. Kuhnel. *Differential Gemoetry: Curves - Surface - Manifolds*. American Mathematical Society., second edition, 2005.

[14] C.B. Macdonald and S.J. Ruuth. Level set equations on surfaces via the Closest Point Method. *J. Sci. Comput.*, 35(2–3), 2008.

[15] C.B. Macdonald and S.J. Ruuth. The implicit Closest Point Method for the numerical solution of partial differential equations on surfaces. *SIAM J. Sci. Comput.*, 2009.

[16] Colin.B. Macdonald. The Closest Point Method for time-dependent processes on surfaces. *Phd thesis, SFU*, 2008.

[17] G. Dal Maso, J. M. Morel, and S. Solimini. A variational method in image segmentation: existence and approximation results. *Acta Matematica*, 168, 1992.

[18] D. Mumford and J. Shah. Optimal approximation by piecewise smooth functions and associated variational problems. *Commun. Pure Appl. Math*, 42, 1989.

[19] S. Osher and J. A. Sethian. Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulation. *J. Comput. Phys.*, 79, 1988.

[20] Richard Prager. 3d Ultrasound using stradx 7.4f. `http://mi.eng.cam.ac.uk/~rwp/stradx/segmentation_menu.html`, 2008.

[21] S.J. Ruuth and B. Merriman. A simple embedding method for solving partial differential equations on surfaces. *J. Comput. Phys.*, 227(3), 2008.

[22] A. Spira and R. Kimmel. Geometric curve flows on parametric manifolds. *J. Comput. Phys.*, 223, 2007.

[23] Z. Tu and S. C. Zhu. Image segmentation by data-driven Markov chain Monte Carlo. *PAMI*, 24, 2002.

[24] H.-K. Zhao, T. Chan, B. Merriman, and S. Osher. A variational level set approach to mulitphase motion. *J. Comput. Phys.*, 127, 1996.