UCLA

COMPUTATIONAL AND APPLIED MATHEMATICS

The Level Set Method Applied to Geometrically Based Motion, Materials Science, and Image Processing (Ph.D. Thesis)

Li-Tien Cheng

June 2000

CAM Report 00-20

Department of Mathematics
University of California, Los Angeles
Los Angeles, CA. 90095-1555

http://www.math.ucla.edu/applied/cam/index.html

University of California
Los Angeles

The Level Set Method Applied to Geometrically Based Motion, Materials Science, and Image Processing

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Mathematics

by

Li-Tien Cheng

2000

To my mom and my dad and my brother

and in memory of the pudge.

Table of Contents

List of Figures

List of Tables

Acknowledgments

I would like to thank my advisor, mentor, and teacher Stanley Osher for his guidance, patience, and support. I am indebted to him for all his help during my graduate studies at UCLA. His energy, intelligence, and diligence are inspiring. I would also like to thank Tony Chan, Russel Caflisch, Bjorn Engquist, Eitan Tadmor, and Heinz-Otto Kreiss for their insights and support in research. I would especially like to further thank Tony Chan for long time guidance and support in all aspects of life.

Other people have also helped me with math questions when I needed help. Thanks goes to Michael Chu, Avery Ching, Albert Ku, Kai Hu, my brother Hsiao-Bing Cheng, and my father Shiu-Yuen Cheng for help in questions in geometry and other pure math subjects. Thanks goes to Ron Fedkiw, Paul Burchard, Barry Merriman, Danping Peng, Steven Ruuth, Shih-Hsien Yu, Yann Brenier, Justin Wan, Peter Blomgren, Hong-Kai Zhao, Chiu-Kwong Wong, Chi-Tien Lin, Myungjoo Kang, Susan Chen, Bo Li, Richard Tsai, and especially Hao-Min Zhou for discussions on various applied math topics.

I also would like to thank my other fellow colleagues for their support during my graduate and undergraduate years. Thanks goes to Chad Sprouse, David Kan, Nataphan Kitisin, Charles Li, Xingwei Hu, Doreen DeLeon, Rachel Caiden, Berta Sandberg, Randolph Cooper, Stan Yoshinobu, Kin Pong Lee, Jonathan Ng, Ron Clark, Hyeseon Shim, Sung Ha Kang, Frederic Gibou, Duc Nguyen, and all my teaching assistants and professors.

Finally, I would like to thank an NDSEG Fellowship for most of the financial support during my graduate studies and NSF DMS 9615854, NSF DMS 9706827, and ONR N00014-97-1-0027 for partial support.

VITA

1976	Born, New York, New York, USA.
1996	B.S. (Mathematics) and M.A. (Mathematics), UCLA.
1996	Daus Scholarship.
1996-1999	NDSEG Fellowship.

Abstract of the Dissertation

The Level Set Method Applied to Geometrically Based Motion, Materials Science, and Image Processing

by

Li-Tien Cheng

Doctor of Philosophy in Mathematics

University of California, Los Angeles, 2000

Professor Stanley Osher, Chair

The level set method has been used successfully in many areas of applied mathematics. We extend its application for geometrically based motion, materials science, and image processing. For geometrically based motion, we study the motions of codimension two objects such as curves in $\mathbf{R}^3$ while allowing merging. We also introduce a level set based representation for constrained problems such as the motion of curves on surfaces. Finally, we compute numerical solutions to the Minkowski Problem using a standard level set approach. Related both to geometrically based motion and materials science, we use a variational based level set method to construct Wulff minimal surfaces through given boundaries. We also run simulations to study a level set method for island dynamics in molecular beam epitaxy. Finally, we modify our algorithm for curves on surfaces to consider image processing of images on surfaces. Along the way, we introduce various applications arising from these methods. Results show that the level set method is very flexible and can easily handle all the problems we look at.

CHAPTER 1

Introduction

We construct and analyze various level set based methods for the topics of geometrically based motion, materials science, and image processing. Numerical results from each method are presented and analyzed.

In Chapter 2, we study a level set representation for moving curves in $\mathbf{R}^3$. Numerical results are given to show that the representation easily allows merging to occur. The method can also handle complicated curves and a variety of different flows such as flow by curvature and flows in the normal and binormal directions.

In Chapter 3, we modify the level set representation for curves in $\mathbf{R}^3$ to study the constrained motion of curves on surfaces. Merging is also automatic here due to the level set representation. Numerical results show that complicated surfaces can be handled with ease as well as complicated curves and a variety of motions. We study motions that correspond to motions of curves in $\mathbf{R}^2$ such as geodesic curvature flow and constant normal flow on surfaces. We also apply the results to compute geodesics and Wulff shapes on surfaces.

In Chapter 4, we apply our method on surfaces to denoise and deblur images painted on surfaces. Once again, complicated surfaces can be considered and standard partial differential equations, for the denoising and deblurring of flat images, especially the total variation based equations, are carried over to surfaces.

We show that the total variation methods on surfaces keep edges sharp, thus producing good results, when denoising or deblurring.

In Chapter 5, we create a level set method to construct numerical solutions arising from the Minkowski Problem. This means for a given function defined on a sphere, we find the ovaloid whose Gauss-Kronecker curvature at a point is given by the function evaluated at the inward normal of the ovaloid at that point. Results show that our method can easily construct a these shapes.

In Chapter 6, we introduce a simple variational based level set method for constructing Wulff minimal surfaces. Standard minimal surfaces are a subset of these. Wulff minimal surfaces are surfaces minimizing the Wulff energy, which is related to crystal shapes, subject to the constraint that the surface passes through given curves. Our method does not require a lot of initial information and can easily deal with complicated curve boundaries.

In Chapter 7, we test a level set method for island dynamics in molecular beam epitaxy. We particularly look at unstable cases in irreversible aggregation. Results show that the method reproduces the instabilities that are inherent in the physical problem.

Proofs of propositions in each chapter are located near the end of the chapter. Figures are also located at the end of each chapter.

CHAPTER 2

The Level Set Representation for Moving Curves in $\mathbf{R}^3$

2.1 Abstract

The level set method [24] was originally designed for problems dealing with codimension one objects, where it has been extremely succesful, especially when merging of the interface occurs. Attempts have been made to modify it for objects of higher codimension. We present numerical simulations of a level set based method for moving curves in $\mathbf{R}^3$ that allows for merging. A vector valued level set function is used with the zero level set representing the curve. Results will show this method can handle many types of curves moving under many types of flows while automatically enforcing merging.

2.2 Introduction

The study of curves in $\mathbf{R}^3$ has many importances, especially under geometrically based motions. These motions include motion of a curve by curvature, torsion, and motions in the normal and binormal direction. These motions can then be applied to phenomena in the physical world. Vortex filaments such as smoke rings can be represented by curves in $\mathbf{R}^3$ and exhibit the merging property, as do the vortex lines in superfluid helium [19]. Curves in $\mathbf{R}^3$ have also been used for active contours in image processing[20]. In geometry, curve shortening is of interest[1] and motion in the binormal direction has a link to the Schrodinger equation. Finally, extension of the level set method as a tool to handle objects of higher codimension is of great interest. A list of other problems dealing with codimension two objects can be found in [14].

2.2.1 Merging

We first clarify merging for curves in $\mathbf{R}^3$. For curves in $\mathbf{R}^2$, the idea is simple and is readily observed in the physical case of two phase flow. For curves in $\mathbf{R}^3$, merging should behave as in the case of smoke rings. This follows curve shortening principles, meaning when two segments of the curve touch, the curve connects in the acute angle directions (see Figure 2.1). Pictures of merging can also be found in [19].

2.2.2 Other Work

Attempts to extend the level set method for use on curves in $\mathbf{R}^3$ have been studied by De Giorgi[11] and Ambrosio and Soner[2]. They were interested in the theoretical aspects of curvature motion and used a single level set function, usually the squared distance to the curve, to represented curves in $\mathbf{R}^3$. This was done in the standard level set way, the curve being represented by the zero level set of the level set function. Note in this formulation, the zero level set is also the set of points achieving the minimum value. One problem with this method is in accurately determining the location of the curve. The main problem, however, for the topic we consider here is that merging does not carry over. A phenomenon called "thickening" occurs, where the zero level set develops a nonempty interior, when curves try to merge (see Figure 2.2). The formulation, however, was successful in determining theoretical aspects of curvature flow in the absence of merging.

The study of curves in $\mathbf{R}^3$ has also been attempted from other directions, for example, using front tracking[13]. This is where the curve is parametrized and numerically represented by discrete points, each of which is then evolved under the flow. Following these points thus gives the curve at all time. The main problem with this approach lies in finding and enforcing merging when it occurs. This has proven to be difficult for curves in $\mathbf{R}^2$ and is equally, if not more so, for curves in $\mathbf{R}^3$. On the other hand, another approach, diffusion generated motion by Ruuth, Merriman, Xin, and Osher[27], can deal with mergings but is limited to curvature flow. The results for curvature flow, however, are good and we have compared them with our results whenever possible.

2.3 Level Set Representation of the Curve

The representation we use makes use of two level set functions to model a curve in $\mathbf{R}^3$, an approach Ambrosio and Soner[2] suggested first but did not pursue because the theoretical aspects were too difficult. In this formulation, a curve is represented by the intersection between the zero level sets of two level set functions, ϕ and ψ, i.e., where $\phi = \psi = 0$. Here, ϕ and ψ can be considered as the two components of a vector valued level set function whose zeros give the curve in $\mathbf{R}^3$. Thus, we can consider arbitrary codimensional objects in arbitrary dimensions by using a vector valued level set function with the right number of components. For example, we would use the zeros of an m component vector valued function over $\mathbf{R}^n$ to represent a codimension m object in dimension n. Note, the sets of points satisfying $\phi = C_1$ and $\psi = C_2$, where C_1 and C_2 are constants, also represent curves in $\mathbf{R}^3$.

In order to move a curve by a certain type of motion, we evolve the functions ϕ and ψ in $\mathbf{R}^3$, keeping in mind that the intersection of their zero level sets gives the curve. Usually, all the other curves gotten from the intersections of level sets of ϕ and ψ will move under the same type of motion.

2.3.1 Geometric Quantities

In order to move a curve by a geometrically based motion, we need to be able to derive all relevant geometric quantities of the curve in terms of our representation. Important quantities include tangent vectors, curvature times normal vectors, normal vectors, binormal vectors, and torsion times normal vectors.

To find the tangent vectors T, we notice that $\nabla\psi \times \nabla\phi$, taken on the curve, is tangent to the curve. So the tangent vectors are just a normalization of this,

$$T = \frac{\nabla\psi \times \nabla\phi}{|\nabla\psi \times \nabla\phi|}.$$

Note if we replace ϕ with $-\phi$, the tangent vectors will be reversed.

For the curvature times normal, κN, of the curve, we use the definition that this is the change in the tangent vector along the curve,

$$\kappa N = \frac{dT}{ds}.$$

Using directional derivatives, this becomes

$$\kappa N = \nabla T \cdot T = \begin{pmatrix} \nabla T_1 \cdot T \\ \nabla T_2 \cdot T \\ \nabla T_3 \cdot T \end{pmatrix}.$$

We may then use the expression for T to write this in terms of ϕ and ψ.

For the normal vectors, N, of the curve, we use the definition

$$N = \frac{\kappa N}{|\kappa N|}.$$

The binormal vectors B are obtained using the definition

$$B = T \times N.$$

Note the binormal vectors are reversed if we replace ϕ by $-\phi$. Finally, the torsion times normal vectors can be derived using the definition

$$\tau N = -\nabla B \cdot T.$$

All these geometric quantities can thus be written in terms of ϕ and ψ by using the corresponding expression for T. Also, note the above geometric quantities derived at an arbitrary point in $\mathbf{R}^3$ are quantities for the curve $\{\phi = C_1, \psi = C_2\}$ that passes through that point.

2.4 The Evolution Equation

Moving the curve in $\mathbf{R}^3$ using our representation requires moving the level set functions ϕ and ψ. We will first investigate the motion of a curve under a given vector field v in $\mathbf{R}^3$. From standard level set theory, we know that the partial differential equation

$$\phi_t + v \cdot \nabla\phi = 0,$$

moves the level sets of ϕ according to v. Similarly,

$$\psi_t + v \cdot \nabla\psi = 0,$$

moves the level sets of ψ according to v. Therefore, the system of partial differential equations

$$\begin{cases} \phi_t + v \cdot \nabla\phi &= 0 \\ \psi_t + v \cdot \nabla\psi &= 0, \end{cases}$$

moves the intersections of the level sets of ϕ and ψ, especially the zero level sets, according to v.

A better derivation can be given. Let $\gamma(s,t)$ denote the intersection between two level sets of ϕ and ψ. So $\phi(\gamma(s,t),t) = C_1$ and $\psi(\gamma(s,t),t) = C_2$. Taking a derivative with respect to t then gives

$$\begin{cases} \nabla\phi(\gamma(s,t),t)\cdot\gamma_t(s,t) + \phi_t(\gamma(s,t),t) &= 0 \\ \nabla\psi(\gamma(s,t),t)\cdot\gamma_t(s,t) + \psi_t(\gamma(s,t),t) &= 0. \end{cases}$$

Since the curve is moving under the vector field v, this means $\gamma_t(s,t) = v$. Therefore, since C_1 and C_2 are arbitrary, we get back the system of equations above, valid in all of $\mathbf{R}^3$.

2.4.1 Other Types of Motions

By allowing v to depend on ϕ and ψ and their derivatives, we can write down the evolution equations for any type of motion. For example, setting $v = \kappa N$ in the evolution equations above gives curvature motion. Similarly, $v = N$ gives motion in the normal direction at unit speed, $v = B$ gives motion in the binormal direction at unit speed, and $v = \tau N$ gives torsion motion. We now study some of these motions more carefully and present numerical discretizations and results.

2.5 Curvature Motion

The evolution equations for curvature motion take the form

$$\begin{cases} \phi_t + \kappa N\cdot\nabla\phi &= 0 \\ \psi_t + \kappa N\cdot\nabla\psi &= 0. \end{cases}$$

This, in fact, can be derived from modified gradient descent minimizing the length of the curve. Notice first that the length of the curve coming from the intersection of the zero level sets of ϕ and ψ can be written as

$$L(\phi, \psi) = \int_{\mathbf{R}^3} \delta(\phi)\delta(\psi)|P_{\nabla\psi}\nabla\phi||\nabla\psi|dx,$$

where δ is the one dimensional delta function and P_v is the orthogonal projection matrix that projects vectors onto the plane with normal vector v. In $\mathbf{R}^3$, we have $|P_v w| = \frac{|v \times w|}{|v|}$ and so we can also write the length as

$$L(\phi, \psi) = \int_{\mathbf{R}^3} \delta(\phi)\delta(\psi)|\nabla\phi \times \nabla\psi|dx.$$

Proposition 2.1 *The Euler-Lagrange equations for this are*

$$\begin{aligned} 0 &= -\nabla \cdot \left(\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|}|\nabla\psi|\right)\delta(\psi)\delta(\phi) \\ 0 &= -\nabla \cdot \left(\frac{P_{\nabla\phi}\nabla\psi}{|P_{\nabla\phi}\nabla\psi|}|\nabla\phi|\right)\delta(\phi)\delta(\psi). \end{aligned}$$

This can be rewritten as

$$\begin{pmatrix} 0 \\ 0 \end{pmatrix} = \begin{pmatrix} \delta(\phi)\delta(\psi) & 0 \\ 0 & \delta(\phi)\delta(\psi) \end{pmatrix} \begin{pmatrix} -\nabla \cdot \left(\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|}|\nabla\psi|\right) \\ -\nabla \cdot \left(\frac{P_{\nabla\phi}\nabla\psi}{|P_{\nabla\phi}\nabla\psi|}|\nabla\phi|\right) \end{pmatrix}.$$

Following level set theory, we try to replace the matrix of delta functions, which we consider as smoothed out delta functions, with a positive definite matrix that will give, on the right hand side of the Euler-Lagrange equations,

$$\begin{pmatrix} \kappa N \cdot \nabla\phi \\ \kappa N \cdot \nabla\psi \end{pmatrix}.$$

Then modified gradient descent minimizing the length of the curve will be equivalent to curvature motion. For this, we have

Proposition 2.2 *The replacement matrix that gives equivalence is*

$$\begin{pmatrix} \frac{|\nabla\phi|}{|P_{\nabla\phi}\nabla\psi|} & \frac{\nabla\phi \cdot \nabla\psi}{|P_{\nabla\psi}\nabla\phi||\nabla\psi|} \\ \frac{\nabla\phi \cdot \nabla\psi}{|P_{\nabla\psi}\nabla\phi||\nabla\phi|} & \frac{|\nabla\psi|}{|P_{\nabla\psi}\nabla\phi|} \end{pmatrix}.$$

This means curvature motion, in our representation, follows a curve shortening process.

For all our numerical discritizations, we lay down a uniform grid over $\mathbf{R}^3$ and use finite difference schemes. The uniform grid simplifies finite difference scheme construction and implementation. We then discretize the curvature evolution equation by using second order central differencing on all spacial derivatives. The case $T = 0$ needs to be regularized to remove singularities in the curvature expression. For the time discretization, we use Total Variation Diminishing Runge-Kutta (TVD-RK) of third order (see [29]). The Courant-Friedrichs-Lewy (CFL) condition says that the time step Δt needs to be less than a constant times Δx^2, where Δx denotes the spacial step size.

This representation allows merging to occur, as seen in numerical simulations. The time of merging does not have to be computed and there is no need for switches to enforce merging. It is automatically handled by the representation. The evolution equation is simply solved at each time step and the resulting ϕ and ψ gives the curve, even when merging has taken place previously. Also the curve location does not have to be computed until the curve is to be plotted.

Plotting itself is carried out by using interpolation schemes. Each cube in the grid is broken up into six tetrahedra, inside of which ϕ and ψ can be approximated by hyperplanes. The intersection between the zero level sets of the two hyperplanes can then be computed, giving a small segment of line inside each tetrahedron. The union of all these segments gives an approximation of the curve.

In Table 2.1, we show the method is second order accurate. This test was done by looking at the evolution of a double helix in $\mathbf{R}^3$, where we know the exact solution.

grid size	error	order
$32 \times 32 \times 32$	0.00459276	
$64 \times 64 \times 64$	0.00140586	1.7079
$128 \times 128 \times 128$	0.000356941	1.9777

Table 2.1: Order of accuracy analysis for a double helix moving under curvature flow. Results show the method is second order accurate.

We consider the motion of a single helix in Figure 2.3. The helix straightens out as time progresses. Evolution of two slightly translated helices is presented in Figure 2.4. The helices move independent of each other, each one straightening itself out, until they touch. Merging then occurs and the resulting curve continues to flow by curvature and shrink. Another example with two helices is shown in Figure 2.5. The two strands again touch at a certain time and merging occurs. The resulting curve then continues to flow by curvature. In Figure 2.6, we consider the motion of linked rings. At first each ring will shrink its radius by a speed of $\frac{1}{r}$, where r is the radius. Eventually the rings touch and merging occurs. The resulting curve then continues to flow by curvature. Finally, Figures 2.7 and 2.8 show the evolution of other curves.

2.6 Normal and Binormal Motion

The evolution equation for motion in the normal direction at unit speed is

$$\begin{cases} \phi_t + N \cdot \nabla\phi &= 0 \\ \psi_t + N \cdot \nabla\psi &= 0. \end{cases}$$

We discretize the time derivative using third order TVD-RK and all space derivatives using second order central differencing. Singularities occurring where $|T| =$

0 and $|\kappa N| = 0$ are regularized. Note geometrically, N is not defined when $|\kappa N| = 0$.

We consider a potato chip curve as our initial curve in Figure 2.9. Normal motion in our simulations causes a kink to develop in the curve after a certain time. This is because parts of the curve have rammed together and merging has been enforced. In a standard tracking algorithm that allows curves to pass through each other, a swallow tail would appear instead.

The evolution equation for motion in the binormal direction at unit speed is

$$\begin{cases} \phi_t + B \cdot \nabla\phi &= 0 \\ \psi_t + B \cdot \nabla\psi &= 0. \end{cases}$$

Once again, we discretize the time derivative using third order TVD-RK and all the space derivatives using second order central differencing. The singularities occurring at $|T| = 0$ and $|\kappa N| = 0$ are regularized.

We consider the simple case of a circle moving under binormal motion in Figure 2.10. The circle is translated, which is the correct solution. In Figure 2.11, we look at the evolution of two helices. Both slightly rotate and in opposite directions.

The cases of normal and binormal motion are not as nice as the case of curvature motion and not all initial curves evolve nicely. More work needs to be done on the discretization of the equations, especially the regularization of singularities. Another flow not studied here is with the velocity field $T \times \kappa N$. This is related to the motion of vortex lines in superfluid helium.

2.6.1 Combinations

We can combine the above motions to form other types of motions. For example, taking the velocity field $v = N + \epsilon\kappa N$ gives motion in the normal direction with some curvature flow.

We look at the evolution of the potato chip curve in Figure 2.12. ϵ is taken to be 0.1 and the result has no kink anymore due to the presence of the curvature term.

2.7 Remarks

Some difficulties that are left include theoretical justification, which we will not investigate here, creating an optimal local method, and initializing ϕ and ψ to create a given curve.

2.7.1 Local Level Set Method

We would like to solve our evolution equations only in a small neighborhood of the curve. This would give optimal efficiency both in speed and memory usage. Solving in all of $\mathbf{R}^3$, however, is sometimes needed, for example, when the curves coming from the intersections of other level sets of ϕ and ψ play a role. It can also be adequate, for example, if the problem we are considering requires other equations to be solved in all of $\mathbf{R}^3$. But for the type of problems we have discussed here, a more local method is needed. Such an algorithm for curves in $\mathbf{R}^2$ has been created[25] but many things need to be added when considering curves in $\mathbf{R}^3$. The main idea involves only doing computations in a tube around the curve with radius a constant times Δx. Reinitialization needs to be performed at certain times to keep errors from the boundary of the tube from influencing the curve.

For curves in $\mathbf{R}^2$, this is accomplished by replacing the level set function with the signed distance function to the curve at each time step.

For our first step in optimizing our algorithm, we can cut down one of the dimensions by localizing around the zero level set of one of the level set functions, say ψ. The case where ψ is fixed in time is considered in Chapter 3 Section 3.12. This algorithm needs to be modified to allow ψ to move, a simple concept that follows standard local level set ideas[25].

A completely localized, and thus optimal, algorithm for curves in $\mathbf{R}^3$, however, has not yet been completed. Certain problems may arise from such an algorithm, for example, a twist of the level set functions about the curve may introduce spurious curves during merging. This kind of twist will not occur if ϕ and ψ are globally defined. Also the reinitialization process in standard local level set methods needs to be further studied and adapted to curves in $\mathbf{R}^3$.

2.7.2 Initialization

Another issue is how to choose the initial level set functions to create a desired curve. In some occasions, the initial functions are given, for example, as Clebsch variables (see [33]). Usually, however, they need to be constructed by hand. The difficulty in this lies in forming the functions ϕ and ψ in all of $\mathbf{R}^3$. Forming these functions only local to the curve is very easy but sometimes these local constructions cannot be extended to $\mathbf{R}^3$, causing problems during merging. However, as Figure 2.6 shows, creating initial level set functions for complicated curves is not impossible. Another problem that may arise is that some constructions may hamper merging. This needs to be studied further.

2.8 Conclusion

We have analyzed a level set based method for representing and moving higher codimensional objects, especially curves in $\mathbf{R}^3$. As numerical results show, the representation automatically handles mergings of the curve. Evolution equations on the level set functions can then used to move the curve under a variety of flows. An underlying uniform grid allows for easy high order finite difference scheme constructions. Thus, we have set up a foundation to deal with higher codimensional objects, especially in the presence of merging.

2.9 Proofs of Propositions

Proof of Proposition 2.1.

We will derive the Euler-Lagrange equations for the energy

$$E(\phi, \psi) = \int_{\mathbf{R}^n} |P_{\nabla\psi}\nabla\phi||\nabla\psi|\delta(\phi)\delta(\psi)dx,$$

where ϕ and ψ are real valued functions over $\mathbf{R}^n$.

Note

$$|P_{\nabla\psi}\nabla\phi|^2|\nabla\psi|^2 = |\nabla\phi|^2|\nabla\psi|^2 - (\nabla\phi\cdot\nabla\psi)^2 = |P_{\nabla\phi}\nabla\psi|^2|\nabla\phi|^2,$$

and, therefore,

$$|P_{\nabla\psi}\nabla\phi||\nabla\psi| = \sqrt{|\nabla\phi|^2|\nabla\psi|^2 - (\nabla\phi\cdot\nabla\psi)^2} = |P_{\nabla\phi}\nabla\psi||\nabla\phi|.$$

So

$$\begin{aligned}
&\frac{d}{ds}(|P_{\nabla\psi+s\nabla\nu}(\nabla\phi+s\nabla\eta)||\nabla\psi+s\nabla\nu|\delta(\phi+s\eta)\delta(\psi+s\nu)) = \\
&= \frac{d}{ds}(|P_{\nabla\psi}(\nabla\phi+s\nabla\eta)||\nabla\psi|\delta(\phi+s\eta)\delta(\psi))+ \\
&\quad \frac{d}{ds}(|P_{\nabla\psi+s\nabla\nu}\nabla\phi||\nabla\psi+s\nabla\nu|\delta(\phi)\delta(\psi+s\nu)) \\
&= \frac{d}{ds}(|P_{\nabla\psi}(\nabla\phi+s\nabla\eta)||\nabla\psi|\delta(\phi+s\eta)\delta(\psi))+ \\
&\quad \frac{d}{ds}(|P_{\nabla\phi}(\nabla\psi+s\nabla\nu)||\nabla\phi|\delta(\psi+s\nu)\delta(\phi)).
\end{aligned}$$

Therefore,

$$\begin{aligned}
&\left.\frac{d}{ds}\right|_{s=0} E(\phi+s\eta,\psi+s\nu) = \\
&\left.\frac{d}{ds}\right|_{s=0} \int_{\mathbf{R}^n} |P_{\nabla\psi}(\nabla\phi+s\nabla\eta)||\nabla\psi|\delta(\phi+s\eta)\delta(\psi)dx+ \\
&\left.\frac{d}{ds}\right|_{s=0} \int_{\mathbf{R}^n} |P_{\nabla\phi}(\nabla\psi+s\nabla\nu)||\nabla\phi|\delta(\psi+s\nu)\delta(\phi)dx.
\end{aligned}$$

Then, from Proposition 3.3, we get that the Euler-Lagrange equations are

$$\begin{aligned}
0 &= -\nabla\cdot\left(\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|}|\nabla\psi|\right)\delta(\psi)\delta(\phi) \\
0 &= -\nabla\cdot\left(\frac{P_{\nabla\phi}\nabla\psi}{|P_{\nabla\phi}\nabla\psi|}|\nabla\phi|\right)\delta(\phi)\delta(\psi).
\end{aligned}$$

Proof of Proposition 2.2.

We will show that we can get $\kappa N\nabla\phi$ and $\kappa N\nabla\psi$ from replacing the values of the $\delta(\phi)\delta(\psi)$ in the right hand sides of the Euler-Lagrange equations.

First we see from Proposition 3.5 that

$$\begin{aligned}
\kappa N\cdot P_{\nabla\psi}\nabla\phi &= -\nabla\cdot\left(\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|}||\nabla\psi|\right)\frac{|P_{\nabla\psi}\nabla\phi|}{|\nabla\psi|} \\
\kappa N\cdot P_{\nabla\phi}\nabla\psi &= -\nabla\cdot\left(\frac{P_{\nabla\phi}\nabla\psi}{|P_{\nabla\phi}\nabla\psi|}||\nabla\phi|\right)\frac{|P_{\nabla\phi}\nabla\psi|}{|\nabla\phi|}.
\end{aligned}$$

So we want to look for functions f and g that may depend on ϕ, ψ, and their derivatives such that

$$\begin{pmatrix} f_1 & g_1 \\ g_2 & f_2 \end{pmatrix} \begin{pmatrix} \kappa N \cdot P_{\nabla\psi}\nabla\phi \\ \kappa N \cdot P_{\nabla\phi}\nabla\psi \end{pmatrix} = \begin{pmatrix} \kappa N \cdot \nabla\phi \\ \kappa N \cdot \nabla\psi \end{pmatrix}.$$

This is equivalent to

$$\begin{aligned} \kappa N \cdot (f_1 P_{\nabla\psi}\nabla\phi + g_1 P_{\nabla\phi}\nabla\psi) &= \kappa N \cdot \nabla\phi \\ \kappa N \cdot (g_2 P_{\nabla\psi}\nabla\phi + f_2 P_{\nabla\phi}\nabla\psi) &= \kappa N \cdot \nabla\psi. \end{aligned}$$

Therefore, we are looking for a decomposition of $\nabla\phi$ in terms of $P_{\nabla\psi}\nabla\phi$ and $P_{\nabla\phi}\nabla\psi$ and similarly for $\nabla\psi$. First note that this is possible since .

$$\nabla\phi \times P_{\nabla\phi}\nabla\psi = \nabla\phi \times \nabla\psi = \nabla\psi \times P_{\nabla\psi}\nabla\phi,$$

so we have that $\nabla\phi, \nabla\psi, P_{\nabla\psi}\nabla\phi, P_{\nabla\phi}\nabla\psi$ all lie on the same plane. Also $P_{\nabla\psi}\nabla\phi$ and $P_{\nabla\phi}\nabla\psi$ span the plane if we disregard the degenerate case where $\nabla\phi$ and $\nabla\psi$ are parallel.

Taking a dot product of $\nabla\phi$ with the equations

$$\begin{aligned} \nabla\phi &= f_1 P_{\nabla\psi}\nabla\phi + g_1 P_{\nabla\phi}\nabla\psi \\ \nabla\psi &= g_2 P_{\nabla\psi}\nabla\phi + f_2 P_{\nabla\phi}\nabla\psi, \end{aligned}$$

gives

$$\begin{aligned} f_1 &= \frac{|\nabla\phi|^2}{|P_{\nabla\psi}\nabla\phi|^2} \\ g_2 &= \frac{\nabla\phi \cdot \nabla\psi}{|P_{\nabla\psi}\nabla\phi|^2}. \end{aligned}$$

Also taking a dot product of $\nabla\psi$ with the same equations gives

$$\begin{aligned} f_2 &= \frac{|\nabla\psi|^2}{|P_{\nabla\phi}\nabla\psi|^2} \\ g_1 &= \frac{\nabla\phi\cdot\nabla\psi}{|P_{\nabla\phi}\nabla\psi|^2}. \end{aligned}$$

Note that $f_1 = f_2$.

Therefore, replacing

$$\begin{pmatrix} \delta(\phi)\delta(\psi) & 0 \\ 0 & \delta(\phi)\delta(\psi) \end{pmatrix},$$

in the Euler-Lagrange equations by

$$\begin{pmatrix} f_1\frac{|P_{\nabla\psi}\nabla\phi|}{|\nabla\psi|} & g_1\frac{|P_{\nabla\phi}\nabla\psi|}{|\nabla\phi|} \\ g_2\frac{|P_{\nabla\psi}\nabla\phi|}{|\nabla\psi|} & f_2\frac{|P_{\nabla\phi}\nabla\psi|}{|\nabla\phi|} \end{pmatrix},$$

will give the curvature flow evolution equation. So the replacement matrix can be written as

$$\begin{pmatrix} \frac{|\nabla\phi|}{|P_{\nabla\phi}\nabla\psi|} & \frac{\nabla\phi\cdot\nabla\psi}{|P_{\nabla\psi}\nabla\phi||\nabla\psi|} \\ \frac{\nabla\phi\cdot\nabla\psi}{|P_{\nabla\psi}\nabla\phi||\nabla\phi|} & \frac{|\nabla\psi|}{|P_{\nabla\psi}\nabla\phi|} \end{pmatrix}.$$

In general in level set theory, this replacement can be used, unchanged, for all other Euler-Lagrange equations of curve flow in $\mathbf{R}^3$.

Note the determinant of our replacement matrix is

$$\frac{|\nabla\phi|^2|\nabla\psi|^2 - (\nabla\phi\cdot\nabla\psi)^2}{|P_{\nabla\psi}\nabla\phi|^2|\nabla\psi|^2},$$

which is equal to 1. Also the first entry is positive and so the matrix is positive definite.

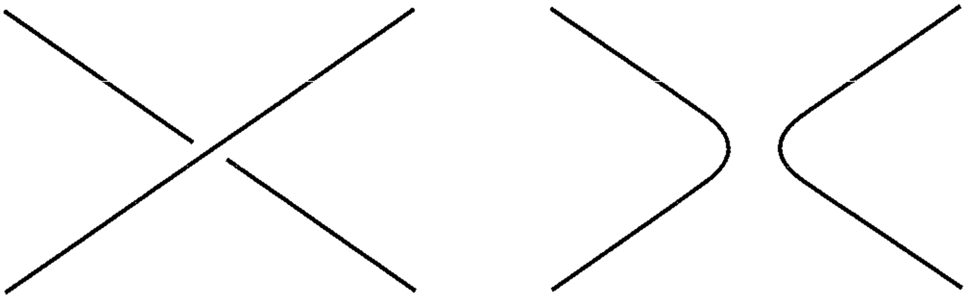

Figure 2.1: The picture on the left shows two lines, one on top of the other. The picture on the right shows our merging requirement in action when the two lines touch. Note the curve reconnects according to the acute angles.

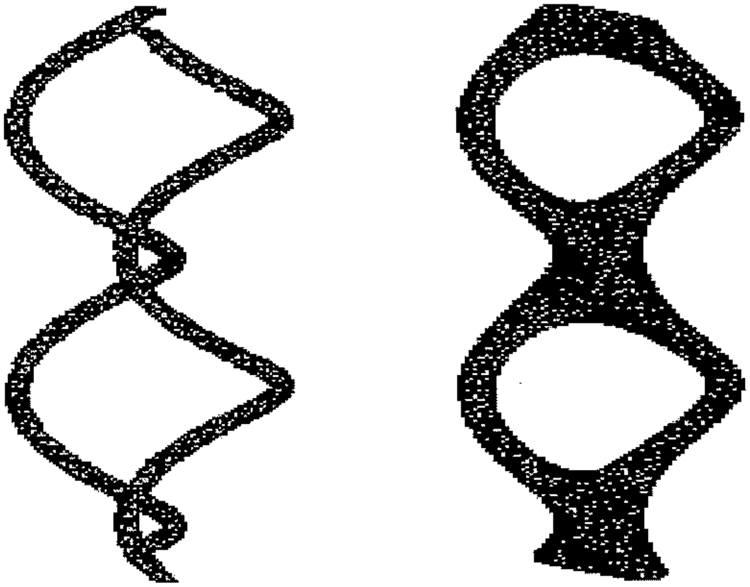

Figure 2.2: These pictures were generated using De Giorgi's method. The picture on the left shows two helices about to merge. The picture on the right shows thickening occurring when the two helices merge.

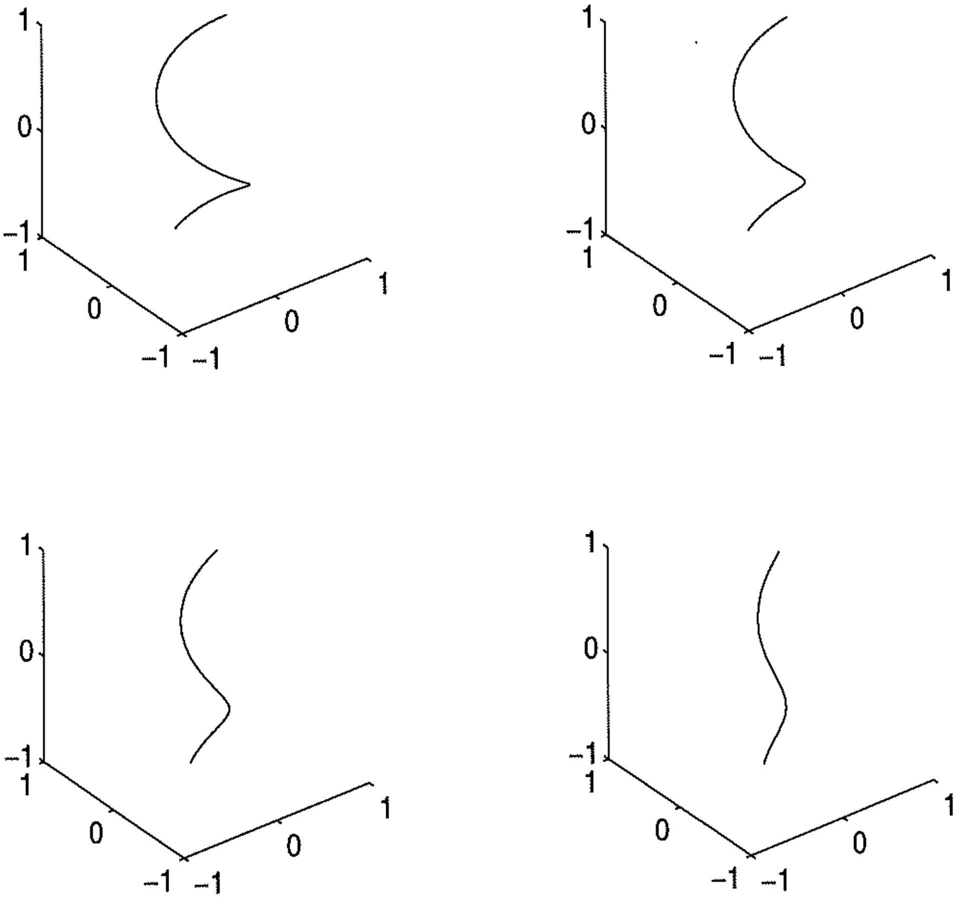

Figure 2.3: This shows a single helix evolving under curvature motion. The curve remains helical but the radius about the center axis shrinks.

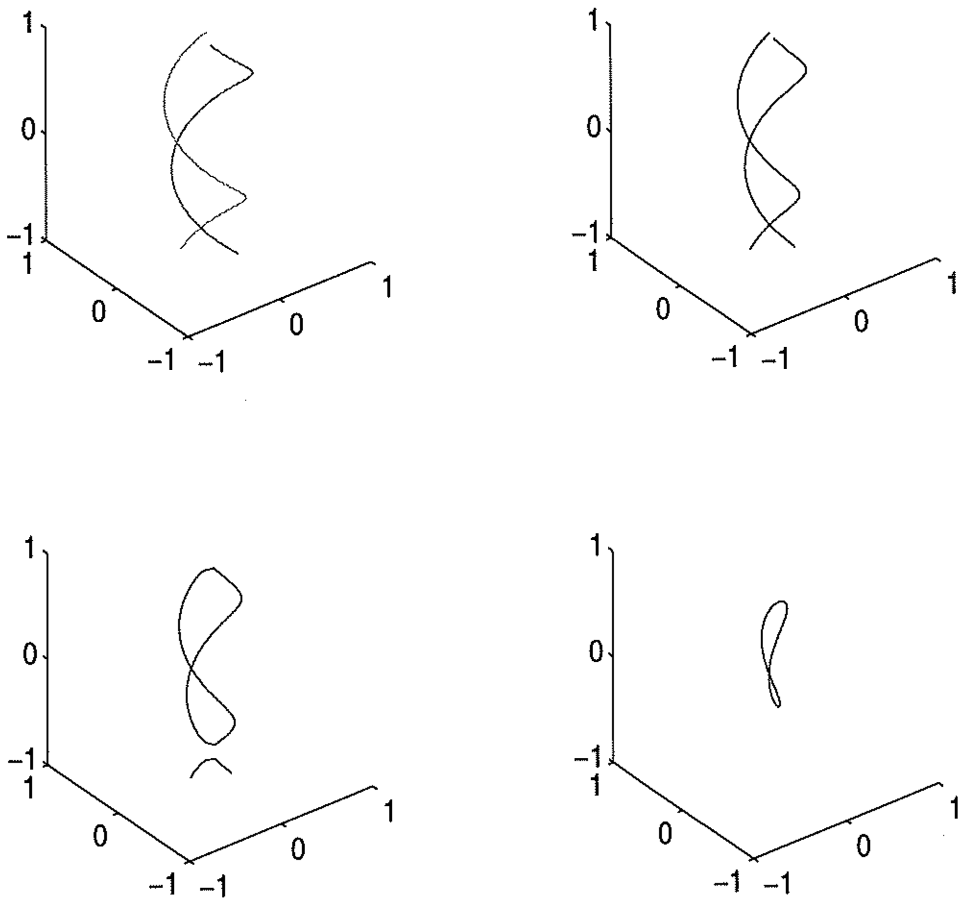

Figure 2.4: This shows two slightly translated helices evolving under curvature motion. The translation allows the helices to touch and merge. The resulting curve then continues to evolve under curvature motion.

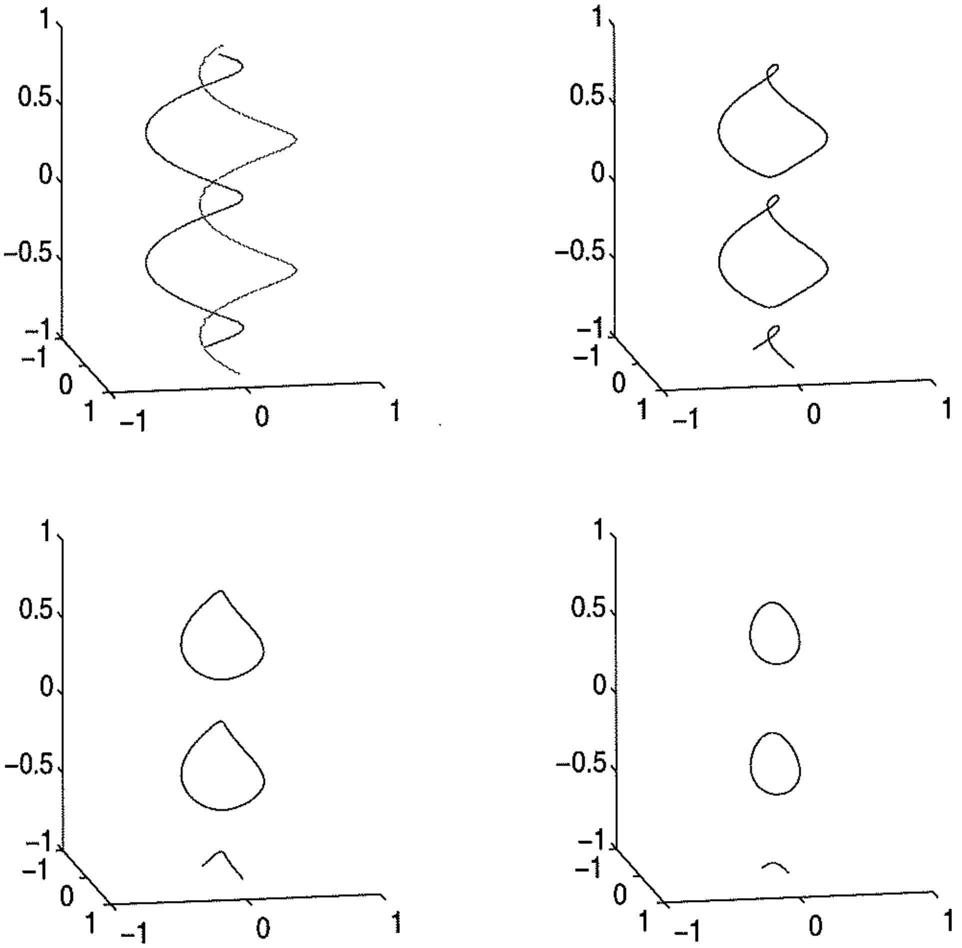

Figure 2.5: This shows another two helices evolving under curvature motion. The two touch and merge at a certain time. The resulting curve then continues to evolve under curvature motion.

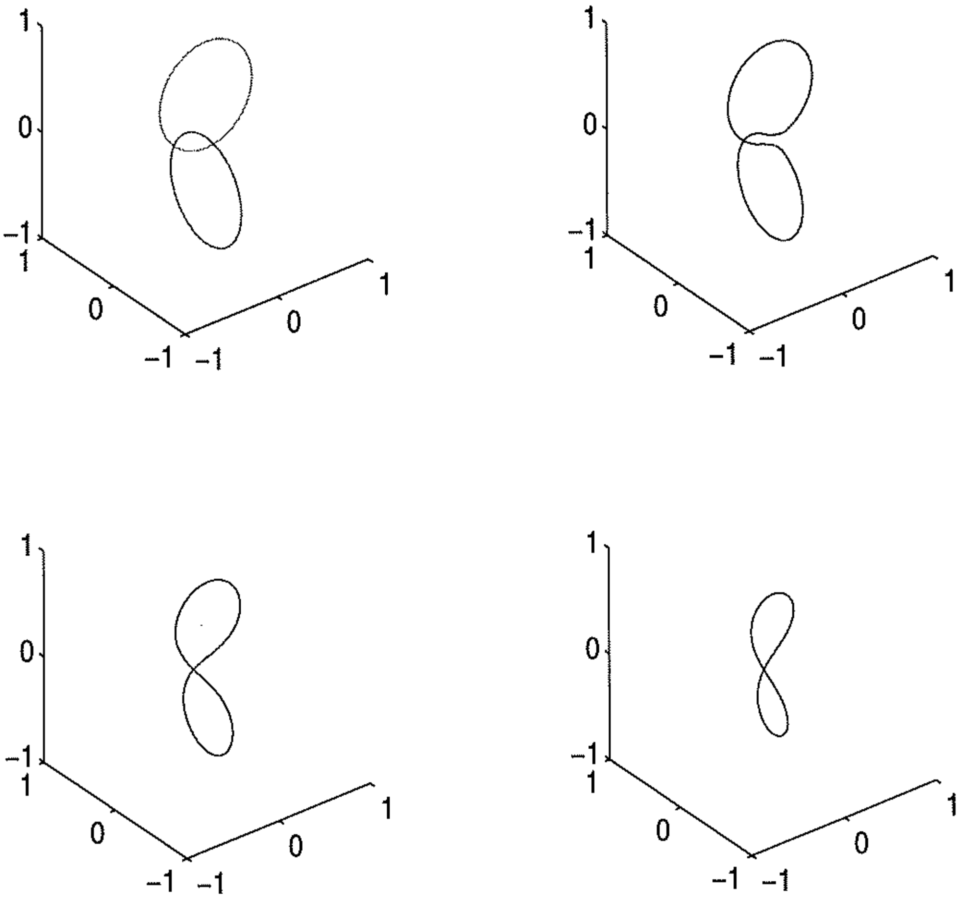

Figure 2.6: This shows two linked rings evolving under curvature motion. The two rings shrink independent of each other until they touch and merge. The resulting curve then continues to evolve under curvature motion.

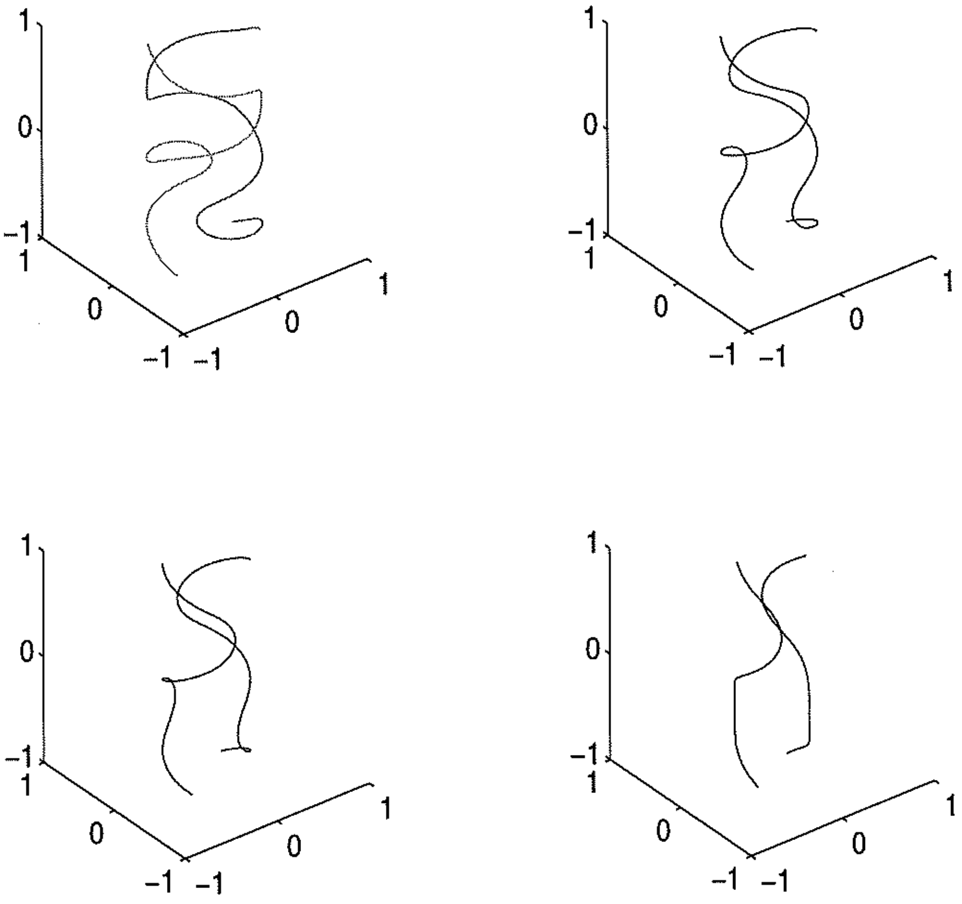

Figure 2.7: This shows two complicated curves evolving under curvature motion. The picture stops before merging occurs.

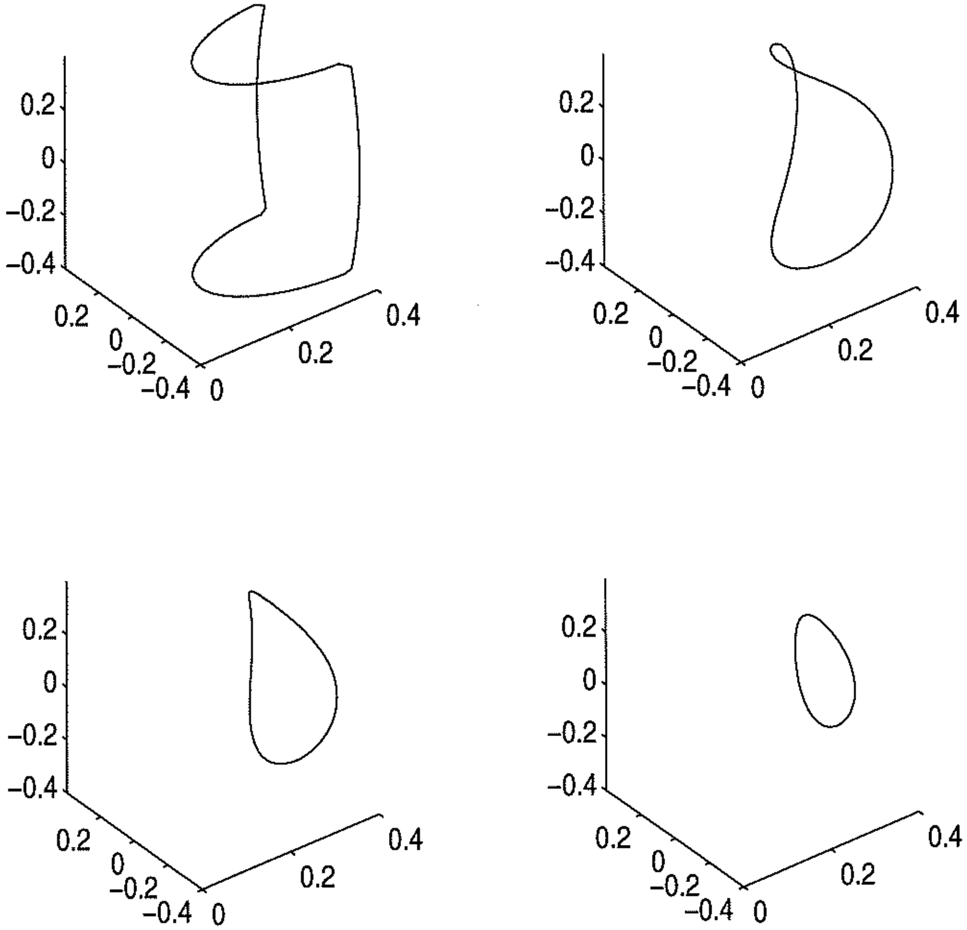

Figure 2.8: This shows a curve with a kink evolving under curvature motion. The kink is smoothed out almost immediately.

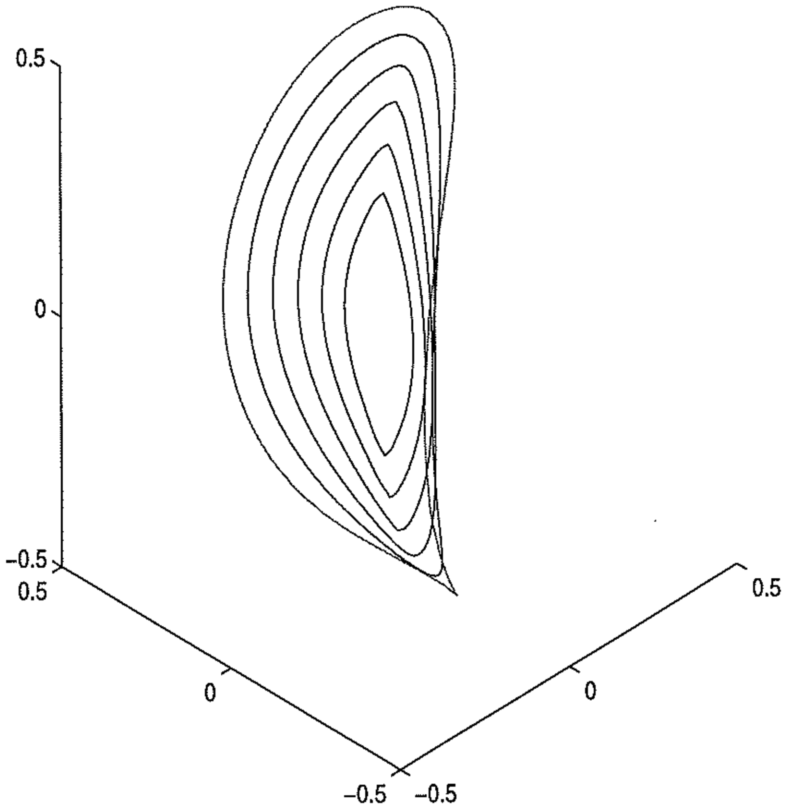

Figure 2.9: This shows the time evolution of a curve under constant flow in the normal direction. The curve is initially shaped like the boundary of a potato chip and shrinks thereafter. Note a kink forms in the curve at a certain time, which is an indication of merging.

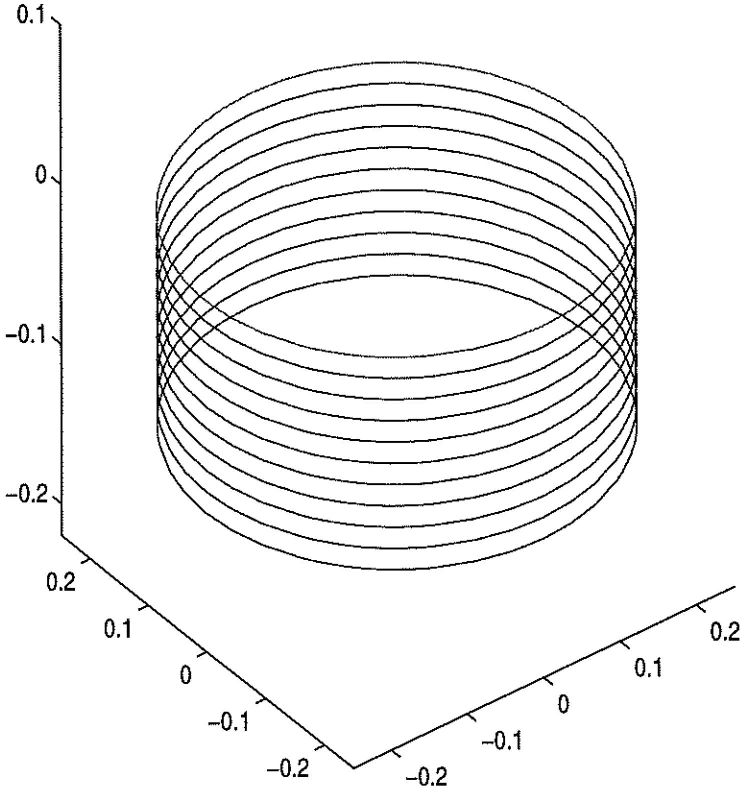

Figure 2.10: This shows the time evolution of a circle under constant flow in the binormal direction. Since the original curve lies on a plane, the evolution is simply translation in the normal direction of the plane. In this picture, the circle is moving in the downward direction.

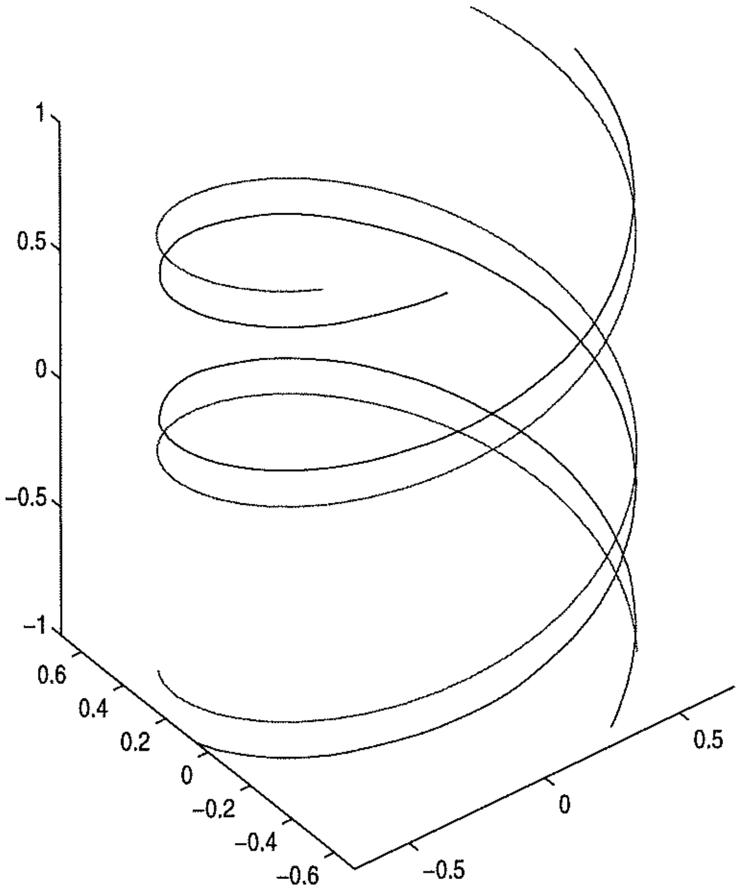

Figure 2.11: This shows the time evolution of two helices under constant flow in the binormal direction. Each helix rotates about its center axis but in opposite directions.

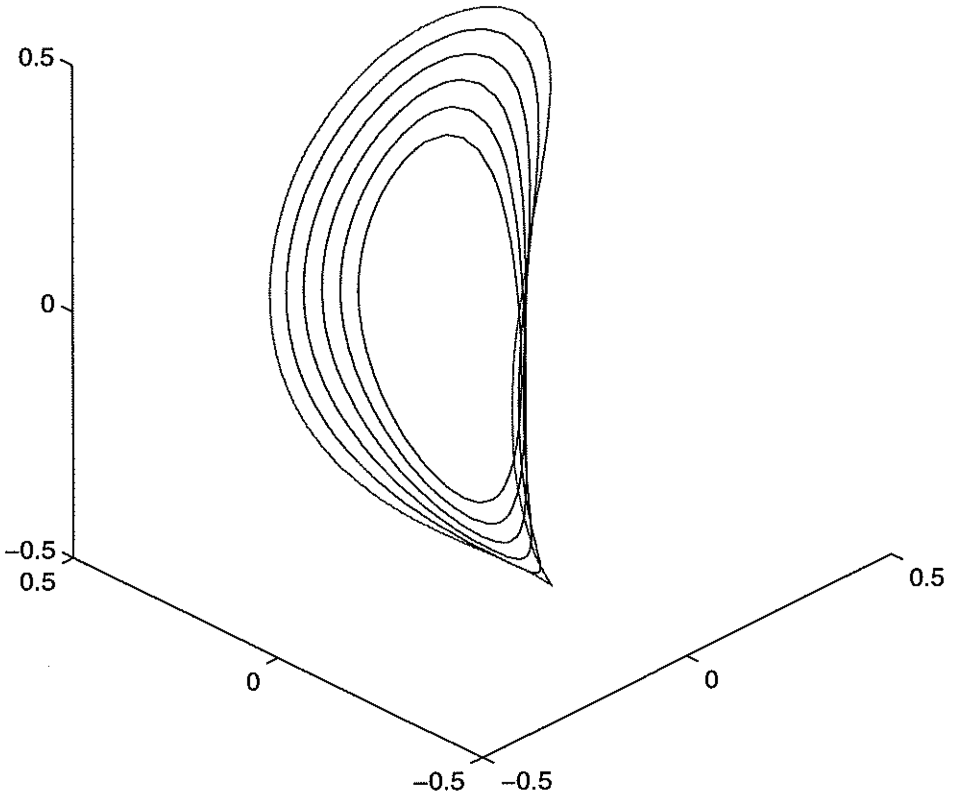

Figure 2.12: This shows the time evolution of an initial potato chip curve by unit speed in the normal direction combined with 0.1 times curvature motion. Note a kink no longer forms due to adding the curvature term.

CHAPTER 3

A Level Set Representation for Moving Curves on Surfaces

3.1 Abstract

Given a surface M in $\mathbf{R}^3$ and a curve γ_0 on M, our goal is to compute the time evolution of this curve constrained to be on the surface under a specified type of motion. In the course of this evolution, one part of the curve may run into another part. When this happens, we require the two parts to merge together. This phenomenon can be found in many physical situations, for instance, in two dimensional two phase flow, where the curve denotes the interface between two fluids. We will create a level set based method for computing geometrically based motions for curves on surfaces under the merging requirement and develop related applications. Results will show that our method works for hypersurfaces which can be represented by the boundary of an open set and can handle all geometrically based curve motions.

3.2 Introduction

On planes, the level set method[24] easily accomplishes all this for a certain class of curves. In this method, $\mathbf{R}^2$ is used to represent the plane. The curve at any time t is then represented by the zero level set at time t of a real valued function ϕ on $\mathbf{R}^2$. This function is called a level set function. The ability of a curve to be represented as the zero level set of a function means the curve must be the boundary of an open set in $\mathbf{R}^2$. This limits the types of curves the level set method can handle but is a very natural setup for the case of two phase flow. The motion of such a curve can then be carried out through the evolution of ϕ, keeping in mind that the zero level set of ϕ at any time gives the curve at that time. Usually the evolution of ϕ is governed by a partial differential equation.

Numerically, a uniform grid is placed on $\mathbf{R}^2$ and finite difference schemes are used on the evolution equation. Efficiency both in memory and speed can still be preserved by only storing data and computing near the front[25], though sometimes at the price of a loss of accuracy. The main advantage to this method is that merging is automatically handled by the representation. The time of merging does not need to be computed and no extra work is needed to enforce merging, unlike front tracking methods. The evolution equation is simply solved, using finite differences, up to the desired time. The curve can then be interpolated from the level set function at the end of a run, when we want to plot the curve. During the run, the curve location is not needed and can remain uncomputed. This ease in handling merging is one of the main reasons for using level set methods. However, even when merging does not occur, level set methods are still attractive because they allow for simple finite difference schemes and so, in the end, are easy to program and use. All this naturally leads us to attempt to extend this method for use on more general surfaces.

3.2.1 Setup

We represent a surface M in $\mathbf{R}^3$ by the zero level set of a real valued function ψ on $\mathbf{R}^3$ and a curve on that surface at time t by the intersection of the zero level set at time t of a real valued function ϕ on $\mathbf{R}^3$ with the zero level set of ψ. We will mainly consider the case where M is static in time which implies ψ does not depend on t. On the other hand, the time evolution of ϕ allows us to follow the moving curve, keeping in mind that the curve at time t is the intersection of the zero level set of ϕ at time t and the zero level set of ψ. Because of our way of representing the surface, only a specific class of surfaces, boundaries of open sets in $\mathbf{R}^3$, can be handled by this method. Similarly, only a specific class of curves on the surface, boundaries of open sets on M, can be considered. This means that there is a notion of the inside and outside of a curve and we take, for definiteness, the inside to be where ϕ is negative and the outside to be where ϕ is positive. Once again, this is especially natural for curves denoting the interface between two fluids on M. We also note that ψ and ϕ need only be defined in a neighborhood of the curve and not necessarily in all of $\mathbf{R}^3$. However, for simplicity we will continue treating them as functions over all of $\mathbf{R}^3$. The only concern will be when we need to slightly modify the method to have optimal efficiency both in speed and memory. This will be discussed later.

Note our setup is basically the same as in Chapter 2 except ψ is now held fixed in time. Thus the constrained problem of moving curves on surfaces turns out to be easier, using our setup, than the unconstrained problem of moving curves in $\mathbf{R}^3$. Also note the method runs the same for any surface M. This is because ψ over $\mathbf{R}^3$ is used instead of M. Thus complicated surfaces are as easily handled as any other surface.

3.2.2 Other Methods

Other methods currently used to study curves on surfaces include work done by Chopp[8], Kimmel[17], and Kimmel and Sethian[18]. Chopp's work is on curve flow under geodesic curvature. His is a level set based method but requires computing on coordinate patches of the manifold projected into $\mathbf{R}^2$. Finding the coordinate patches can be complicated for general manifolds. Kimmel also studied geodesic curvature flow and applied his results to image processing of images on surfaces. His algorithm, however, only handles surfaces that can be represented by the graph of a function. Kimmel and Sethian's work is on finding geodesic paths on manifolds. The method is level set based but the surface needs to be triangulated and it seems to be only for surfaces that are graphs of functions.

Our method, however, can handle more general hypersurfaces and curves moving under a variety of motions, as well as be extended for other applications. We mention that Bertalmio, Sapiro, and Randall's work[3] is of the same flavor but is only for region tracking. We rederive and extend their results in Section 3.12.

Because our method can extend all the motions of curves in $\mathbf{R}^2$ to general surfaces, the applications of these can also be carried over. This includes work on two phase flow, vortex motion, crystal growth, island dynamics for molecular beam epitaxy, image processing, and a variety of geometrically based motions (see [22]). We, however, will mainly consider geometrically based motions here. We now develop various notation and tools for our representation to help simplify and clarify future calculations.

3.3 Preliminary

Given a vector w in $\mathbf{R}^3$, let P_w be the orthogonal projection matrix defined by

$$P_w = I - \frac{w \otimes w}{|w|^2},$$

where I is the identity matrix. Thus the components of the matrix are

$$(P_w)_{ij} = \delta_{ij} - \frac{w_i w_j}{|w|^2},$$

where δ_{ij} is the Kronecker delta function. Note for x in M and ν the normal vector in $\mathbf{R}^3$ of M at x, P_ν projects vectors onto M at x, i.e., P_ν projects vectors onto the tangent plane of M at x. Now for X a vector field in $\mathbf{R}^3$ we define the differential operator $P_X\nabla$ by its components,

$$(P_X\nabla)_i = \sum_{j=1}^{3} \left(\delta_{ij} - \frac{X_i X_j}{|X|^2} \right) \partial_{x_j}.$$

Note this is just the projection matrix P_X multiplying the gradient vector operator ∇. So given a real valued function u on $\mathbf{R}^3$,

$$(P_X\nabla)u = P_X\nabla u,$$

and given a vector field Y on $\mathbf{R}^3$,

$$P_X\nabla \cdot Y = \sum_{i=1}^{3} (P_X\nabla)_i Y_i.$$

We will constantly use this notation with the vector field $X = \nabla\psi$, which is parallel at each point in $\mathbf{R}^3$ to the normal of the level set surface of ψ that passes through the point. So given a point x in R^3, $P_{\nabla\psi}$ projects vectors onto the level set surface of ψ passing through x. Therefore, if $x \in M$, $P_{\nabla\psi}$ will project vectors onto M at x. This is very useful for putting vector fields onto the surface. Note especially that $P_{\nabla\psi}\nabla u$, evaluated on M, is the projection of the gradient vector

∇u in $\mathbf{R}^3$ onto M. This turns out to be equivalent to the surface gradient of u on M. Similarly, $P_{\nabla\psi}\nabla \cdot X$, evaluated on M, is equivalent to the surface divergence of X on M. We now present a few useful properties of our notation.

Proposition 3.1 *Let v, w, z be vectors, X a vector field, and u a real valued function all in $\boldsymbol{R}^3$. Also let e_i denote the ith vector of the standard orthonormal basis of $\boldsymbol{R}^3$. Then we have the following identities:*

(a) $P_w v \cdot z = v \cdot P_w z = P_w v \cdot P_w z.$

(b) $(P_X \nabla)_i u = \nabla u \cdot P_X e_i.$

(c) $P_{\nabla u}\nabla \cdot (P_{\nabla u} X) = \nabla \cdot (P_{\nabla u} X |\nabla u|)\frac{1}{|\nabla u|}.$

3.3.1 Linking $\mathbf{R}^2$ Equations to Equations on Surfaces

In the course of studying the motion of curves on surfaces, we will need to study partial differential equations on surfaces. Usually, from work done using the original level set method, we already know the form of the partial differential equation corresponding to the same type of motion for curves in $\mathbf{R}^2$. We would like to put this equation onto a surface M, hopefully preserving its important properties.

Given a point x on M, we will write the form of the equation on the surface at this point. Let ν be the normal vector of M at x and let $\tilde{e}_1, \tilde{e}_2, \tilde{e}_3$ be an orthonormal basis of $\mathbf{R}^3$ with $e_3 = \nu$. We also let $\tilde{\partial}_i$ be the derivative corresponding to $\tilde{e}_i$, $i = 1, 2, 3$. We can then write the partial differential equation on M at x by treating the tangent plane at x as $\mathbf{R}^2$, since the form of the equation is known there. This means we put all quantities in the $\mathbf{R}^2$ equation onto the tangent plane at x. This just involves changing those quantities to fit the new frame $\tilde{e}_1$ and $\tilde{e}_2$.

Note this will especially involve the surface gradient vector operator defined by

$$\nabla^S u = \tilde{\partial}_1 u \tilde{e}_1 + \tilde{\partial}_2 u \tilde{e}_2,$$

for u a function on M and

$$\nabla^S \cdot X = \langle \tilde{\partial}_1 X, e_1 \rangle + \langle \tilde{\partial}_2 X, e_2 \rangle,$$

for X a vector field on M. For example, on M and at x, the Laplacian of a function u takes the form $\tilde{\partial}_1 \tilde{\partial}_1 u + \tilde{\partial}_2 \tilde{\partial}_2 u$, which can be written as $\nabla^S \cdot \nabla^S u$. So in this example, ∇ is replaced by ∇^S to get from the $\mathbf{R}^2$ Laplacian to the surface Laplacian. We will use this procedure to put other partial differential equations in $\mathbf{R}^2$ onto surfaces. Assuming that the important properties of the equations are preserved during this transition, this is a quick way to get the evolution equation we want on M.

The main replacement when putting $\mathbf{R}^2$ partial differential equations onto surfaces is changing ∇ to ∇^S. The connection between the surface gradient ∇^S and our previous operator $P_{\nabla\psi}\nabla$ is given by

Proposition 3.2 *We have the following properties:*

(a) For u a real valued function in $\boldsymbol{R}^3$,

$$\nabla^S u = P_{\nabla\psi} \nabla u,$$

on M, where $\nabla^S u$ means the surface gradient applied to the restriction of u on M.

(b) For X a vector field in $\boldsymbol{R}^3$ which is tangent to M on M,

$$\nabla^S \cdot X = P_{\nabla\psi} \nabla \cdot X,$$

on M, where $\nabla^S \cdot X$ means the divergence of the restriction of X on M with respect to the surface gradient.

So ∇^S and $P_{\nabla\psi}\nabla$ are equivalent on M. The difference is that $P_{\nabla\psi}\nabla$ is easier to deal with numerically. Because of this, we will write all our equations using this form. We also note the importance of replacing the integral in $\mathbf{R}^2$, $\int_{\mathbf{R}^2} dx$, with the surface integral, $\int_S dA$, which is equivalent to $\int_{\mathbf{R}^3} \delta(\psi)|\nabla\psi|dx$. Finally, the $\mathbf{R}^2$ equation should be invariant under a rotation of frames in $\mathbf{R}^2$. Otherwise, writting it on the surface may not be a well defined process.

3.4 General Numerics

The main advantage of our approach lies in the effective numerical schemes that can be used. In general, we lay down a uniform grid in $\mathbf{R}^3$. In reality, not all the points in this grid need to be used since we only have to solve the partial differential equation in a neighborhood of our curve. This is called a local level set method, which we will discuss later. The level set functions ψ and ϕ are either given or created on this grid initially. Usually we use analytical expressions for ψ but we may just as well give values of ψ only at points in the grid. The partial differential equation for ϕ is then solved by using appropriate finite difference schemes, which the uniform grid lets us easily create and implement. Note also under our representation, the curve will not leave the surface and so the constraint of the curve being on the surface is always satisfied. In fact, the curve location does not need to be determined for our computations but only when we plot the curve. The level set method representation also automatically takes care of any merging that may occur. The partial differential equations for the evolution are just solved in the same way until the end time regardless of whether merging has occurred or not. This ease in handling merging is one of the main reasons for using level set methods.

3.5 Introduction to Flows

In the following sections, we will use our format to generate and solve evolution equations for curves on surfaces moving under constant normal flow, geodesic curvature flow, Wulff flow, and flow under fixed enclosed surface area. In the process, we will develop other uses for these flows such as obtaining signed distance functions, geodesics, Wulff minimal curves, and Wulff shapes. Finally, we extend our results to allow the surface to also move. We will mostly derive the evolution equations in a few ways, by putting an $\mathbf{R}^2$ equation onto the surface, by finding a velocity field under which to move the curve, and sometimes through modified gradient descent minimizing an energy. The first way is quick and easy but the other ways are more geometric and intuitive.

3.6 Flow Under A Given Velocity Field

We first consider the simple problem of moving a curve on a surface by a given and fixed velocity field tangent to the surface. The first step is to extend all our quantities to $\mathbf{R}^3$, creating ψ from the surface, an initial ϕ from the curve, and v from the velocity field. There are various numerical methods that can do this, e.g., see [6]. The evolution for ϕ then becomes

$$\phi_t + P_{\nabla\psi} v \cdot \nabla\phi = 0,$$

which means we are moving the level sets of ϕ in $\mathbf{R}^3$ under the velocity field $P_{\nabla\psi} v$. The projection matrix in front of v keeps each level set of ψ independent from the others so that the flow on one level set of ψ will not affect or be affected by the flow on the rest of them. So under this velocity field, for a given level set surface of ψ, the level sets of ϕ on that surface will move according to the velocity field projected onto that surface and, especially, the zero level set of ϕ

on the zero level set surface of ψ, i.e. the curve on M, will move according to v on M. This means the evolution equation gives flow on M under the velocity field v, which is what we want.

Another way to derive this is to look at the surface $\{\psi = C_2\}$ and the curve on that surface $\gamma(s,t)$ obtained from the intersection of $\{\phi = C_1\}$, taken at time t, with the surface. We study the flow of γ on this surface according to a vector field that is tangent to the surface, $P_{\nabla\psi}v$. Considering general C_1 and C_2 allows us to obtain an evolution equation valid in all of $\mathbf{R}^3$. From the definition of γ, we have $\phi(\gamma,t) = C_1$ for all s and t. Therefore, taking a derivative with respect to t gives

$$\nabla\phi(\gamma,t)\cdot\gamma_t + \phi_t(\gamma,t) = 0.$$

The curve moves under the vector field $P_{\nabla\psi}v$ implies that $\gamma_t = P_{\nabla\psi(\gamma)}v(\gamma)$. Therefore, the form of the equation becomes

$$\phi_t(\gamma,t) + P_{\nabla\psi(\gamma)}v(\gamma)\cdot\nabla\phi(\gamma,t) = 0.$$

So, on the curve, we have

$$\phi_t + P_{\nabla\psi}v\cdot\nabla\phi = 0.$$

C_1 and C_2 arbitrary then imply that this equation is valid in all of $\mathbf{R}^3$, giving us back the same equation as above.

The derived evolution equation is a partial differential equation of Hamilton Jacobi form and can be numerically solved using TVD-RK of third order in time and Hamilton Jacobi Weighted Essentially Non-Oscillatory method (WENO) of fifth order in space using the Godunov scheme[15]. The associated CFL condition says that Δt, the time step, must be less than a constant times Δx, the spacial step, with the constant depending on the magnitude of v. Also, the singularity arising from $|\nabla\psi| = 0$ needs to be regularized. This can be accomplished, for example, by replacing $|\nabla\psi|$ with $\sqrt{\psi_{x_1}^2+\psi_{x_2}^2+\psi_{x_3}^2+\epsilon^2}$, where ϵ is positive and small, when it appears in a denominator.

The above process can also be used to derive evolution equations for more general flows. First, a valid velocity field v, which now may depend on ϕ and its derivatives, must be derived. This will depend on the type of flow being considered. Then the evolution equation will take the same form as above,

$$\phi_t + P_{\nabla\psi}v \cdot \nabla\phi = 0.$$

This equation will move the level sets of ϕ in $\mathbf{R}^3$ under the wanted velocity field and thus will move the zero level set of ϕ on M according to the flow being considered. It is also valid in more space dimensions, where ψ and ϕ are real valued functions on $\mathbf{R}^n$ and the projection matrix is an n by n matrix. Note we cannot use the above discretization anymore for general v. The valid discretization of the equation will depend on the form of v, for example, if $-P_{\nabla\psi}v\cdot\nabla\phi$ is elliptic, then we can use central differencing schemes. We will constantly use this velocity field process to derive and validate the evolution equations for our flows.

Note for curves in $\mathbf{R}^2$ and using the original level set method, the evolution equation looks like

$$\phi_t + v \cdot \nabla\phi = 0,$$

where v is a velocity field given in $\mathbf{R}^2$. We want to look at the form of this partial differential equation on the surface M. Given x on M, note $\nabla\psi$ is normal to M at x and let $\tilde{e}_1, \tilde{e}_2, \tilde{e}_3$ be an orthonormal basis in $\mathbf{R}^3$ with $\tilde{e}_3 = \nabla\psi$ at x. This frame then allows us to define the surface gradient operator ∇^S at x as before and so the equation on the surface will take the form

$$\phi_t + v \cdot \nabla^S\phi = 0,$$

or, in detail,

$$\phi_t + \langle v, \tilde{e}_1 \rangle \tilde{\partial}_1 \phi + \langle v, \tilde{e}_2 \rangle \tilde{\partial}_2 \phi = 0.$$

This can be rewritten in the usual format,

$$\phi_t + v \cdot P_{\nabla\psi} \nabla\phi = 0,$$

which is equivalent to what we got previously. So writting the $\mathbf{R}^2$ equation on the surface also gives the correct evolution equation.

3.7 Constant Normal Flow

In this problem, we would like to evolve a curve in the outward normal direction at a constant speed C on the surface. This means at time t, the curve we are looking for is the set of points of distance Ct, measured on the surface, away from γ_0 in the outward direction. Note that moving inward corresponds to C being negative.

We first use our approach for writting $\mathbf{R}^2$ equations on the surface to quickly generate the evolution equation. The corresponding evolution equation for curves in $\mathbf{R}^2$, under the original level set method, is

$$\phi_t + C|\nabla\phi| = 0.$$

Once again, given x on M, let $\tilde{e}_1, \tilde{e}_2, \tilde{e}_3$ be an orthonormal basis in $\mathbf{R}^3$ with $\tilde{e}_3 = \nabla\psi$ at x. This allows us to define ∇^S at x and so the evolution equation on the surface takes the form

$$\phi_t + C|\nabla^S\phi| = 0,$$

or, in detail,

$$\phi_t + C\sqrt{(\tilde{\partial}_1\phi)^2 + (\tilde{\partial}_2\phi)^2}.$$

This can then be rewritten as

$$\phi_t + C|P_{\nabla\psi}\nabla\phi| = 0,$$

which is the correct equation. We will, however, verify that it indeed moves a curve in the outward normal direction by speed C by rederiving it using the more intuitive velocity field approach.

In this approach, we want to calculate a velocity field v under which the level sets of ϕ, and especially the zero level set, will move in the correct manner. For fixed t, consider the surface $\{\psi = C_1\}$ and the curve generated by intersecting this surface with $\{\phi = C_2\}$, where C_1 and C_2 are constants. Note the case we are interested in is $C_1 = C_2 = 0$ but by considering arbitrary C_1 and C_2, we get a velocity field valid in all of $\mathbf{R}^3$ which can be used to evolve ϕ in $\mathbf{R}^3$. Now, on this curve, v should be normal to the curve, have length C, and be tangent to the surface. Such a v will give the desired motion for the curve on the surface. From this, we deduce

$$v = C\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|}.$$

Note, we can use vector cross products instead, since we are in $\mathbf{R}^3$, along with the identity

$$\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|} = \frac{\nabla\psi \times \nabla\phi}{|\nabla\psi \times \nabla\phi|},$$

to rewrite our expressions but we will stick with the more general form. Also, if $C = 1$, note v becomes the outward normal of the curve on the surface.

Under such a velocity field, the evolution equation for ϕ takes the form

$$\phi_t + v \cdot \nabla\phi = 0,$$

since $P_{\nabla\psi}v = v$. Simplifying, we get

$$v \cdot \nabla\phi \;=\; C\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|} \cdot \nabla\phi$$

$$\begin{aligned} &= C\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|}\cdot P_{\nabla\psi}\nabla\phi \\ &= C|P_{\nabla\psi}\nabla\phi|. \end{aligned}$$

So the evolution equation for moving curves on surfaces by constant normal flow is

$$\phi_t + C|P_{\nabla\psi}\nabla\phi| = 0,$$

or, using vector cross products,

$$\phi_t + C\frac{|\nabla\psi \times \nabla\phi|}{|\nabla\psi|} = 0.$$

Note if we have a partial differential equation of the above form, even with C depending on ϕ and its derivatives, then we can say the curve will move by speed C in the normal direction. In fact, all evolution equations for flows can be written in this form. This is because given a velocity field v tangent to the level set surfaces of ψ, then at each point x, we can decompose v in terms of $\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|}$ and vectors perpendicular to it. Thus $v\cdot P_{\nabla\psi}\nabla\phi$ is equal to $C|P_{\nabla\psi}\nabla\phi|$, for some C. So moving under the vector field v is the same as moving in the normal direction by speed C.

The partial differential equation we derived with C constant is of Hamilton-Jacobi form and so we discretize it using Hamilton-Jacobi WENO of fifth order along with Local Lax-Friedrichs (LLF) in space and TVD-RK of third order in time. To satisfy the CFL condition, Δt needs to be smaller than a constant times Δx. The term $|P_{\nabla\psi}\nabla\phi|$ also needs to be regularized to remove the singularity arising from $|\nabla\psi| = 0$.

We can also study the behavior of this flow in higher dimensions. The evolution equation

$$\phi_t + C|P_{\nabla\psi}\nabla\phi| = 0,$$

is still valid with ψ and ϕ real valued functions over the space $\mathbf{R}^n$. When we drop a dimension and flow points on curves, the evolution equation takes the form

$$\phi_t + C\frac{|\psi_y\phi_x - \psi_x\phi_y|}{|\nabla\psi|} = 0.$$

Note the numerator of the second term is the absolute value of the Jacobian of ψ and ϕ.

In Table 3.1, we see that the method is second order accurate before merging occurs. This was checked for a circle moving on a sphere by unit normal flow, i.e., flow in the normal direction at unit speed. The whole algorithm, including the second order accurate plotter, is included in this test. By using a higher order plotter, we can get higher order accuracy.

grid size	error	order
$32 \times 32 \times 32$	0.00561706	
$64 \times 64 \times 64$	0.001703	1.7217
$128 \times 128 \times 128$	0.000447336	1.9286
$256 \times 256 \times 256$	0.000115601	1.9522

Table 3.1: This is the order of accuracy analysis for unit normal flow. The example considered was a circle moving on a sphere. The results show second order accuracy. This is because the accuracy of the whole algorithm, including the second order interpolation scheme used to plot the curve, is tested.

In Figure 3.1, we show a curve moving over two mountains. The curve breaks into two pieces and each piece moves up each mountain. In Figure 3.2, we show a curve moving on a volcano. The curve starts out outside the volcano and goes up and into it. In Figure 3.3, we show a curve on a two holed torus. The curve moves across the two holed torus and breaks and merges multiple times. Thus we see that the motion of the curve by constant normal flow on complicated surfaces, even when merging occurs, is easily taken care of by our algorithm. Finally, we show in Figure 3.4 flow in the normal direction by a non-constant speed. For each point on the curve, this speed is equal to a function β evaluated at the outward normal of the curve. The function we chose is $beta(x) = |x_1| + |x_2| + |x_3|$, which is related to crystal shapes. Note the squarish aspect of the growing curve.

3.8 Signed Distance Function

In an extension of constant normal flow, we would like to find the signed distance away from a curve γ confined to a surface M. The signed distance on M of a point away from a curve is the minimal distance measured on the surface, with a negative sign if the point lies inside the curve, of that point away from the points of the curve. We solve the problem by once again setting ψ to have M as its zero level set and trying to find a real valued function d in $\mathbf{R}^3$ such that given a point $x \in M$, $d(x)$ gives the signed distance of x away from γ. We call d a signed distance function of γ on M. Note this problem is different from the ones we have previously studied because we are looking for a function defined on all of M.

For γ a curve in $\mathbf{R}^2$, finding the signed distance function using the original level set method is accomplished by introducing a time element and creating a partial differential equation whose steady state solution values give signed distance. Starting with a level set function ϕ initially having γ as its zero level set and negative inside γ, the equation

$$\phi_t + \operatorname{sgn}(\phi(x,0))(|\nabla\phi| - 1) = 0,$$

will give signed distance as its steady state solution. The signum function keeps

$\phi = 0$ on γ for all time and the rest of the equation tries to force $|\nabla\phi| = 1$, making the steady state solution a signed distance function. We will derive the correct evolution equation on the surface in two ways, by looking at this equation written on the surface and by using the philosophy behind the equation to recreate it on the surface.

In writting the equation on the surface M, we fix x on M and $\tilde{e}_1, \tilde{e}_2, \tilde{e}_3$ an orthonormal basis of $\mathbf{R}^3$ with $\tilde{e}_3 = \nabla\psi$ at x. Then ∇^S is defined at x and the equation takes the form

$$\phi_t + sgn(\phi(x,0))(|\nabla^S\phi| - 1) = 0,$$

or, in detail,

$$\phi_t + sgn(\phi(x,0))\left(\sqrt{(\tilde{\partial}_1\phi)^2 + (\tilde{\partial}_2\phi)^2} - 1\right) = 0.$$

This can then be written as

$$\phi_t + sgn(\phi(x,0))(|P_{\nabla\psi}\nabla\phi| - 1) = 0.$$

This is the correct equation but we will rederive it in the more detailed and intuitive way by following the basic philosophy of the $\mathbf{R}^2$ equation.

To find d on M, we imitate the method for curves in $\mathbf{R}^2$, i.e., we introduce a time element and create a partial differential equation that has d as its steady state solution on M. Let ϕ initially be such that the intersection of its zero level set with M is γ, with ϕ negative inside γ. If ϕ is a signed distance function on the surface, then it must satisfy $|P_{\nabla\psi}\nabla\phi| = 1$ on M, i.e., the steepest ascent directions of ϕ on M have unit length. So we create an evolution equation such that on M, the steady state solution satisfies this property while keeping the zero level set of ϕ fixed at its original position. One such candidate is

$$\phi_t + sgn(\phi(x,0))(|P_{\nabla\psi}\nabla\phi| - 1) = 0.$$

The steady state viscosity solution on M of this evolution equation will be d. Note that the evolution equation is solved in all of space but steady state may sometimes only be achieved at M.

This equation is also valid in space dimensions other than three and, in fact, the equation for distance on curves in $\mathbf{R}^2$ takes the form,

$$\phi_t + sgn(\phi(x,0)) \left(\frac{\sqrt{|\nabla\psi|^2|\nabla\phi|^2 - (\nabla\psi \cdot \nabla\phi)^2}}{|\nabla\psi|} - 1 \right) = 0.$$

In $\mathbf{R}^3$, written with vector cross products, the equation becomes

$$\phi_t + sgn(\phi(x,0)) \left(\frac{|\nabla\psi \times \nabla\phi|}{|\nabla\psi|} - 1 \right) = 0.$$

The signed distance evolution equation is of Hamilton-Jacobi form and we solve it using Hamilton-Jacobi fifth order WENO-LLF in space and third order TVD-RK in time. We also replace the signum function by a smooth version and regularize to remove the singularity occurring at $|\nabla\psi| = 0$. To satisfy the CFL condition, Δt needs to be less than some constant multiple of Δx.

In Table 3.2, we see that the algorithm for finding distance functions is first order accurate. This is because the curve is moved slightly during iterations of the method. Theoretically this should not happen but because of our signum function and because of the grid, numerically we do get a small shift. Table 3.3 shows that the method has a high order of accuracy when looking at the quantity $||P_{\nabla\psi}\nabla\phi| - 1|$. This means the method gives functions with nice derviatives.

In Figure 3.5, we show a curve on a volcano and the contours of the distance function. Note the contours are well-spaced. In Figure 3.6, we show a curve on a torus and the contours of the distance function. Once again, the contours are well-spaced. Thus we see that the algorithm, though only first order accurate, has accurate derivatives which means it is an accurate signed distance function for a slightly perturbed curve.

grid size	error	order
$32 \times 32 \times 32$	0.020416	
$64 \times 64 \times 64$	0.0106933	0.9330
$128 \times 128 \times 128$	0.00526517	1.0222
$256 \times 256 \times 256$	0.00261509	1.0096

Table 3.2: This is the order of accuracy analysis for the distance function. The curve considered was a circle on a sphere. The results show first order accuracy.

grid size	error	order
$32 \times 32 \times 32$	0.000527837	
$64 \times 64 \times 64$	$4.73339e-06$	6.8011
$128 \times 128 \times 128$	$4.75352e-08$	6.6377

Table 3.3: This is the order of accuracy analysis for the distance function, measuring $||P_{\nabla\psi}\nabla\phi|-1|$. The curve considered was a circle on a sphere. The results show high order accuracy.

3.8.1 Keeping Level Set Functions Well Behaved During Flows

One important application for signed distance functions is the role it can play in keeping the level sets of a function well behaved on the surface during a flow. This helps reduce numerical inaccuracies that may appear from an overly steep or flat level set function. For curves in $\mathbf{R}^2$, this is accomplished by making the level set function into the signed distance function to its zero level set at each time step of the flow. We can do the same for level set functions on surfaces. Note since ψ can be chosen to be well behaved or made so by replacing it by the signed distance function in $\mathbf{R}^3$ to its zero level set surface, we will only study the effect that different ϕ have and assume ψ is well behaved.

Certain types of flows may result in a bunching of level sets, where the function restricted on the surface is steep, or a spreading out of level sets, where the function is almost flat. Numerically, this is undesirable and may introduce large errors in the finite difference approximations. Further errors may also be introduced when using interpolation to find the location of the curve, especially if the function on the surface is almost flat. Finally, flatness also may cause problems if we need to divide by the magnitude of the surface gradient, as is done in geodesic curvature flow. But if the level set function is constrained to be a signed distance function, then the surface gradient will have a magnitude of value 1 everywhere except at kinks. This makes the level set function well behaved, especially near the curve. When we consider a particular flow, i.e., solve an evolution equation for ϕ, the signed distance constraint is enforced usually by iterating the above partial differential equation a few times after every time step of the flow. We only need to iterate a few times since usually only the information around the curve affects its motion and so we only need to enforce signed distance in a neighborhood of the curve. Note the zero level set of ϕ remains fixed when iterating to a signed distance function and so this process theoretically will not affect the flow of the curve on the surface.

We digress here to talk more about keeping ϕ well behaved during its flow. Another way computations may break down is when the level sets of ϕ become tangent to the surface. Note this does not have to do with the level sets of ϕ on the surface, where the signed distance constraint makes ϕ well behaved, but with the behavior of the level sets of ϕ off the surface. For example, $\phi = x_2 - Cx_1$ is already a signed distance function on the surface $x_2 = 0$ for all $C > 0$ but as C tends to zero, the level sets of ϕ become tangent to the surface. This means the surface gradient will be close to zero. Also any small perturbations may greatly shift the location of the curve or even introduce spurious curve parts. To prevent this from happening, we want to make the level set surfaces of ϕ perpendicular to M on M, especially near the curve. This can be accomplished by iterating a few steps of the partial differential equation

$$\phi_t + sgn(\psi)\frac{\nabla\psi}{|\nabla\psi|} \cdot \nabla\phi = 0,$$

at each step of the flow, but after signed distance is enforced. Note this equation forces $\frac{\nabla\psi}{|\nabla\psi|} \cdot \nabla\phi = 0$ in steady state so that the level sets of ϕ will be perpendicular to the surface. It also keeps the level sets of ϕ fixed on the surface so that signed distance on the surface is preserved. The fast marching method can also be used in place of the partial differential equation to create a faster algorithm.

The evolution equation is of Hamilton Jacobi form and we compute it using fifth order WENO-Godunov in space and third order TVD-RK in time. The CFL condition says Δt needs to be less than a constant multiple of Δx.

3.8.2 Geodesics

The signed distance function can also be used to compute geodesics on a surface from points to a curve. This means given a curve γ on M, we want to find the shortest path on M from any point on M to γ. This can be accomplished using a signed distance function d of γ on M. In fact, the shortest path is simply the part of the integral curve of the vector field $-dP_{\nabla\psi}\nabla d$ drawn from the chosen point to γ. This simply means following the steepest descent direction of d on M from the chosen point with speed d. The speed is thus zero at γ and so we follow the integral curve until convergence. The integral curves, $y(s)$, of our vector field can be computed according to the ordinary differential equation

$$\dot{y}(s) = -d(y(s))P_{\nabla\psi(y(s))}\nabla d(y(s)).$$

Note $y(s) \in \mathbf{R}^3$ for all s. For a chosen point x on M, the geodesic from x to γ is thus found by solving the above ordinary differential equation with intial condition $y(0) = x$. This can be done numerically using a Runge-Kutta scheme.

When we want the geodesic between two points a and b, we can first get a $\tilde{d}$ which gives the signed distance function to a small curve around the point a, i.e., approximating a. Then $d = \tilde{d} + \tilde{d}(a)$ is an approximate signed distance function to a on the surface which is exact when the small curve approximating a is at a uniform distance away from a. Using this d in the ordinary differential equation above along with the initial condition $x(0) = b$ allows us to calculate an approximate geodesic. Or, we can require that the distance is given initially in a neighborhood of the point a and then solve for a signed distance function d to a on M by iterating the corresponding evolution equation but only outside the neighborhood of initial given values, the given values being fixed. We can then use this d along with $x(0) = b$ for our initial condition to calculate geodesics.

A drawback of this signed distance function method for geodesics is that when there are two or more geodesics, we have almost no control over which one is chosen. Another minor drawback is that numerical approximations of the geodesics are not forced to lie on the surface in the same way as in our standard representation for curves on surfaces that we have discussed.

In Figure 3.7, we show a curve in the core of a volcano and the geodesics from certain points to that curve. In Figure 3.8, we show a curve on a torus and the geodesics from certain points to that curve. Thus we see that the signed distance function can be used to find geodesics from points to curves on general surfaces.

3.9 Geodesic Curvature Flow

We now consider the problem of moving a curve γ_0 by geodesic curvature on a surface M. This can be accomplished in a few ways, all of which lead to the same evolution equation. The first way is by writting the corresponding evolution equation for curves in $\mathbf{R}^2$ onto the surface. The second way involves finding the curvature times normal vectors of the curve in $\mathbf{R}^3$ and projecting them onto the surface. This gives the velocity vectors with which to move the curve. The third way is studying modified gradient descent minimizing the length of the curve constrained on the surface. The fact that these are all equivalent means moving a curve by curvature is a minimization of the length of the curve.

First Way: Writting $\mathbf{R}^2$ Equation On Surface.

We note that the corresponding evolution equation in $\mathbf{R}^2$ takes the form

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

where $-\nabla \cdot \left(\frac{\nabla \phi}{|\nabla \phi|} \right)$ is the mean curvature of the curve. Given x on M and an orthonormal basis $\tilde{e}_1, \tilde{e}_2, \tilde{e}_3$ in $\mathbf{R}^3$ with $\tilde{e}_3 = \nabla \psi$ at x, we can define ∇^S at x. So the partial differential equation put onto M at x takes the form

$$\phi_t = \nabla^S \cdot \left(\frac{\nabla^S \phi}{|\nabla^S \phi|} \right) |\nabla^S \phi|,$$

where, in fact, $-\nabla^S \cdot \left(\frac{\nabla^S \phi}{|\nabla^S \phi|} \right)$ is the geodesic curvature of the curve. In detail, the equation is

$$\phi_t = \left(\tilde{\partial}_1 \left(\frac{\tilde{\partial}_1 \phi}{\sqrt{(\tilde{\partial}_1 \phi)^2 + (\tilde{\partial}_2 \phi)^2}} \right) + \tilde{\partial}_2 \left(\frac{\tilde{\partial}_2 \phi}{\sqrt{(\tilde{\partial}_1 \phi)^2 + (\tilde{\partial}_2 \phi)^2}} \right) \right) \sqrt{(\tilde{\partial}_1 \phi)^2 + (\tilde{\partial}_2 \phi)^2},$$

with

$$-\tilde{\partial}_1 \left(\frac{\tilde{\partial}_1 \phi}{\sqrt{(\tilde{\partial}_1 \phi)^2 + (\tilde{\partial}_2 \phi)^2}} \right) - \tilde{\partial}_2 \left(\frac{\tilde{\partial}_2 \phi}{\sqrt{(\tilde{\partial}_1 \phi)^2 + (\tilde{\partial}_2 \phi)^2}} \right),$$

the geodesic curvature. We then rewrite all this as

$$\phi_t = \nabla \cdot \left(\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|}|\nabla\psi| \right) \frac{|P_{\nabla\psi}\nabla\phi|}{|\nabla\psi|},$$

with

$$-\nabla \cdot \left(\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|}|\nabla\psi| \right) \frac{1}{|\nabla\psi|}$$

the geodesic curvature. This equation translates to moving a curve on M in the normal direction by geodesic curvature, which is what we want.

Second Way: Projection of Free Space Curvature Times Normal Vector.

Consider the surface $\{\psi = C_1\}$ and the curve generated by intersecting this surface with $\{\phi = C_2\}$, C_1 and C_2 constants. This means $\nabla\psi \times \nabla\phi$ taken on the curve is parallel to the tangent vector of the curve. So the tangent vector of the curve is $T = \frac{\nabla\psi \times \nabla\phi}{|\nabla\psi \times \nabla\phi|}$. Now the curvature times normal vector of the curve in $\mathbf{R}^3$, κN, is the change in the tangent vector along the curve. Therefore, using directional derivatives, we get

$$\kappa N = (\nabla T_1 \cdot T, \nabla T_2 \cdot T, \nabla T_3 \cdot T),$$

where $T = (T_1, T_2, T_3)$. We now project this onto the surface to get $P_{\nabla\psi}\kappa N$. Using this as our velocity field leads to the evolution equation

$$\phi_t = -P_{\nabla\psi}\kappa N \cdot \nabla\phi.$$

This equation also gives geodesic curvature motion of curves on surfaces.

Third Way: Energy Minimization. We consider the energy

$$E(\phi) = \int_{\mathbf{R}^3} \delta(\phi)\delta(\psi)|P_{\nabla\psi}\nabla\phi||\nabla\psi|dx,$$

which gives the length of the curve that comes from the intersection between the zero level sets of ϕ and ψ.

Proposition 3.3 *The Euler-Lagrange equation of this energy is*

$$0 = -\nabla \cdot \left(\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|} |\nabla\psi| \right) \delta(\psi)\delta(\phi).$$

Replacing $\delta(\psi)\delta(\phi)$, which we treat as smoothed out delta functions, by $\frac{|P_{\nabla\psi}\nabla\phi|}{|\nabla\psi|}$ in our gradient descent gives the evolution equation

$$\phi_t = \nabla \cdot \left(\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|} |\nabla\psi| \right) \frac{|P_{\nabla\psi}\nabla\phi|}{|\nabla\psi|},$$

which is exactly what we got using the first method. Note because everything is in $\mathbf{R}^3$, we can also write this equation as

$$\phi_t = \nabla \cdot \left(\frac{(\nabla\psi \times \nabla\phi) \times \nabla\psi}{|\nabla\psi \times \nabla\phi|} \right) \frac{|\nabla\psi \times \nabla\phi|}{|\nabla\psi|^2}.$$

We made the above replacement because it equates our equation with the equation derived using the first method. Also, from standard level set theory, we see that $\delta(\psi)\delta(\phi)$ should be replaced by a quantity that will equate gradient descent for minimizing the enclosed surface area of γ with inward normal flow at unit speed. The enclosed surface area for our curve on M is given by $\int_{\mathbf{R}^3} H(-\phi)\delta(\psi)|\nabla\psi|dx$, where ψ is static and H is the Heaviside function. So the Euler-Lagrange equation is

$$0 = -\delta(\phi)\delta(\psi)|\nabla\psi|,$$

and gradient descent with the above replacement gives

$$\phi_t - \frac{|\nabla\psi \times \nabla\phi|}{|\nabla\psi|} = 0,$$

which is inward normal flow at unit speed.

Equivalence.

We will now show that the evolution equations of the first and third methods are equivalent to the evolution equation of the second method. The main part lies in the following property,

Proposition 3.4 $\nabla \cdot (\tau \times \nabla\psi) = \kappa N \cdot (\nabla\psi \times \tau)$.

Using this to expand the right hand side of the evolution equation in the second method, we get

$$\begin{aligned}
-P_{\nabla\psi}\kappa N \cdot \nabla\phi &= -P_{\nabla\psi}\nabla\phi \cdot \kappa N \\
&= -\kappa N \cdot \left(\frac{|\nabla\psi|^2\nabla\phi - (\nabla\phi \cdot \nabla\psi)\nabla\psi}{|\nabla\psi|^2}\right) \\
&= -\kappa N \cdot \left(\frac{(\nabla\psi \times \nabla\phi) \times \nabla\psi}{|\nabla\psi|^2}\right) \\
&= -\kappa N \cdot \left(\frac{(\nabla\psi \times \nabla\phi) \times \nabla\psi}{|\nabla\psi \times \nabla\phi|}\right) \frac{|\nabla\psi \times \nabla\phi|}{|\nabla\psi|^2} \\
&= -\kappa N \cdot (\nabla\psi \times \tau)\frac{|\nabla\psi \times \nabla\phi|}{|\nabla\psi|^2} \\
&= -\nabla \cdot (\tau \times \nabla\psi)\frac{|\nabla\psi \times \nabla\phi|}{|\nabla\psi|^2} \\
&= \nabla \cdot \left(\frac{(\nabla\psi \times \nabla\phi) \times \nabla\psi}{|\nabla\psi \times \nabla\phi|}\right) \frac{|\nabla\psi \times \nabla\phi|}{|\nabla\psi|^2},
\end{aligned}$$

which is the right hand side of the evolution equation in the third method. This means all the evolution equations are equivalent. We summarize this result in the following.

Proposition 3.5

$$P_{\nabla\psi}\kappa N \cdot \nabla\phi = -\nabla \cdot \left(\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|}|\nabla\psi|\right)\frac{|P_{\nabla\psi}\nabla\phi|}{|\nabla\psi|}.$$

Thus we have

Proposition 3.6 *The evolution equation is degenerate parabolic.*

So we use central differencing in space with third order TVD-RK in time to numerically solve the evolution equation. We also regularize the equation to remove the singularities arising at $|\nabla\psi| = 0$ and $|P_{\nabla\psi}\nabla\phi| = 0$. To satisfy the CFL condition, Δt needs to be less than a constant multiple of Δx^2.

In Table 3.4, we see that the method is second order accurate. This result was obtained by studying a circle moving by geodesic curvature flow on a sphere.

grid size	error	order
$32 \times 32 \times 32$	0.00203138	
$64 \times 64 \times 64$	0.000540219	1.9108
$128 \times 128 \times 128$	0.00014037	1.9443

Table 3.4: This is the order of accuracy analysis for geodesic curvature of a circle on a sphere. The result shows second order accuracy.

In Figure 3.10, we show a curve moving on two mountains. The curve needs to move over the mountains before it can shrink to a point and disappear. In Figure 3.11, we show a curve moving on a bent plane. Note the surface has a kink in it. The curve navigates over this without any problems. In Figure 3.12, we show a curve on a cylinder. The curve evolves and wraps tight around the cylinder, forming a circle. This is a geodesic curve for the surface. So our method can be used to find geodesic curves on surfaces. Altogether, we see that our algorithm can easily perform geodesic curvature flow for curves on surfaces.

3.10 Wulff Flow

We now consider the problem of evolving a curve by Wulff flow on a surface. This means minimizing the energy $\int_\gamma \beta(\nu_M)ds$, where $\beta\colon S^2 \to (0,\infty)$ and ν_M is the unit normal of γ lying on the surface M. Note when $\beta \equiv 1$, we are in the case of geodesic curvature flow. We make a homogeneous degree one extension of β to $\mathbf{R}^3$ and then rewrite the energy using our usual representation to get

$$E(\phi) = \int_{\mathbf{R}^3} \beta\left(\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|}\right)\delta(\psi)\delta(\phi)|\nabla\psi \times \nabla\phi|dx.$$

We call this the Wulff energy.

Proposition 3.7 *The Euler-Lagrange equation of this energy is*

$$0 = -\nabla \cdot (P_{\nabla\psi}\nabla\beta(P_{\nabla\psi}\nabla\phi)|\nabla\psi|)\delta(\psi)\delta(\phi).$$

So the evolution equation, enacting the usual replacement, can be written as

$$\phi_t = \nabla \cdot (P_{\nabla\psi}\nabla\beta(P_{\nabla\psi}\nabla\phi)|\nabla\psi|)\frac{|P_{\nabla\psi}\nabla\phi|}{|\nabla\psi|}.$$

Proposition 3.8 *The partial differential equation is degenerate parabolic if*

$$D^2\beta\left(\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|}\right)$$

is nonnegative definite.

This moves a curve by Wulff flow on a surface.

To see this is the same as writting the $\mathbf{R}^2$ evolution equation on the surface, we note that for curves in $\mathbf{R}^2$, Wulff flow is given by

$$\phi_t = \nabla \cdot \nabla\beta(\nabla\phi)|\nabla\phi|.$$

So given x on M and $\tilde{e}_1, \tilde{e}_2, \tilde{e}_3$ an orthonormal basis for $\mathbf{R}^3$ with $\tilde{e}_3 = \nabla\psi$ at x, we can define ∇^S at x and, thus, the equation on the surface takes the form

$$\phi_t = \nabla^S \cdot \nabla^S\beta(\nabla^S\phi)|\nabla^S\phi|,$$

or, in detail,

$$\phi_t = (\tilde{\partial}_1(\tilde{\partial}_1\beta(\tilde{\partial}_1\phi\tilde{e}_1 + \tilde{\partial}_2\phi\tilde{e}_2)) + \tilde{\partial}_2(\tilde{\partial}_2\beta(\tilde{\partial}_1\phi\tilde{e}_1 + \tilde{\partial}_2\phi\tilde{e}_2)))\sqrt{(\tilde{\partial}_1\phi)^2 + (\tilde{\partial}_2\phi)^2}.$$

This can then be written as

$$\phi_t = P_{\nabla\psi}\nabla \cdot (P_{\nabla\psi}\nabla\beta(P_{\nabla\psi}\nabla\phi))|P_{\nabla\psi}\nabla\phi|,$$

which is equivalent to the equation derived using energy minimization. Note higher dimensions can also be considered by using this evolution equation.

We numerically solve the evolution equation using second order central differencing on all spacial derivatives. The time derivative is discretized using TVD-RK of third order. The equation is also regularized at the singularities that occur at $|\nabla\psi| = 0$ and $|P_{\nabla\psi}\nabla\phi| = 0$.

In Figure 3.13, we show a curve moving on the bottom of a paraboloid. The Wulff energy we use is with a smoothed out version of $\beta(x) = |x_1| + |x_2| + |x_3|$. Its exact form is

$$\beta(x) = \sqrt{x_1^2 + \epsilon^2} + \sqrt{x_2^2 + \epsilon^2} + \sqrt{x_3^2 + \epsilon^2},$$

with $\epsilon = 0.1$. Thus the curve develops a squarish shape while shrinking. In Figure 3.14, we show a curve moving on a bent plane. The curve once again develops a squarish shape and we see that kinks in the surface are not a problem for our algorithm. So altogether, we see that our algorithm can handle Wulff flow of curves on surfaces.

3.10.1 Wulff Minimal Curves

The evolution equation for Wulff flow can be slightly altered to give a method for finding Wulff minimal curves on surfaces. Given a set of points on M, we want to find the curve on the surface that passes through these points with the minimum Wulff energy. We will call the given points boundary points. Details about Wulff minimal surfaces in $\mathbf{R}^3$ are given in Chapter 6.

For $\beta \equiv 1$, we are searching for the curve on the surface of minimal length that passes through the boundary points. When M is a plane, this curve is the boundary of the convex hull of those points. For general surfaces in $\mathbf{R}^3$, the curve will be piecewise geodesics. We find the solution to this problem by solving to steady state the zero level set of ϕ on M in the evolution equation

$$\phi_t = -\mu P_{\nabla\psi}\kappa N \cdot \nabla\phi,$$

where μ is smooth with $\mu(x) = 0$ if x is a boundary point and $\mu(x) = 1$ outside a small neighborhood of the boundary points. The initial curve, γ_0, is chosen to pass through the boundary points.

In the case where the boundary points consist of just two points, a and b, and γ_0 is chosen carefully, we get the geodesic between a and b. However, if the initial curve γ_0 is not chosen carefully, parts of it may merge at later time and not evolve into what we want.

For general β, the evolution equation we are interested in is

$$\phi_t = \mu\nabla \cdot (P_{\nabla\psi} D\beta(P_{\nabla\psi}\nabla\phi)|\nabla\psi|)\frac{|P_{\nabla\psi}\nabla\phi|}{|\nabla\psi|}.$$

We note that when M is a plane, the solution is usually still the boundary of the convex hull of the boundary points. Numerically, the evolution equation is solved using the same finite difference schemes as in the Wulff flow case. The μ is just treated as a coefficient in front of the rest of the equation. For higher dimensions, the same evolution equation holds since it is already in its general form.

Creating a γ_0, or the corresponding initial level set function, that passes through the boundary points may not be easy but sometimes we can simply take as γ_0 any curve on the surface that encompasses all the boundary points. This also makes the initial ϕ easy to construct. When we run the evolution equation in time, the curve will shrink and sometimes end up going through all the boundary points. Other more robust interpolating methods can also be used.

3.11 Fixed Enclosed Surface Area

We now consider the problem of evolving under a certain motion a curve γ on a surface M with the constraint that the surface area of the part of the surface enclosed by γ is fixed in time. We will mainly look at geodesic curvature flow and will comment periodically on more general motions. In this case, the energy involving the length of the curve coupled with the constraint gives us the energy we are interested in. The constraint can be translated as the condition that $\int_{\mathbf{R}^3} H(-\phi)\delta(\psi)|\nabla\psi|dx$ remains constant throughout time. So the new energy to consider is

$$E(\phi) = \int_{\mathbf{R}^3} \delta(\psi)\delta(\phi)|P_{\nabla\psi}\nabla\phi|dx - \lambda \int_{\mathbf{R}^3} H(-\phi)\delta(\psi)|\nabla\psi|dx,$$

where λ is a Lagrange multiplier.

For other flows, we can replace the first integral with the energy corresponding to the type of flow. This just means we are coupling a different energy with the constraint. For example, if we want Wulff flow then we use the Wulff energy. Details of this will be given in the when we discuss Wulff shapes.

The Euler-Lagrange equation then becomes

$$0 = -\nabla \cdot \left(\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|}|\nabla\psi| \right) \delta(\psi)\delta(\phi) + \lambda|\nabla\psi|\delta(\psi)\delta(\phi).$$

Under our usual replacement for $\delta(\psi)\delta(\phi)$ and previous results, we get the evolution equation

$$\phi_t + \lambda|P_{\nabla\psi}\nabla\phi| = \nabla \cdot \left(\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|}||\nabla\psi| \right) \frac{|P_{\nabla\psi}\nabla\phi|}{|\nabla\psi|}.$$

We can find the value of λ by enforcing the constraint,

$$0 = \frac{d}{dt}\int_{\mathbf{R}^3} H(-\phi)\delta(\psi)|\nabla\psi|dx$$

$$
\begin{aligned}
&= \int_{\mathbf{R}^3} \phi_t \delta(\phi)\delta(\psi)|\nabla\psi| dx \\
&= \int_{\mathbf{R}^3} \left(\nabla \cdot \left(\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|} |\nabla\psi| \right) - \lambda |P_{\nabla\psi}\nabla\phi| \right) \delta(\phi)\delta(\psi)|\nabla\psi| dx.
\end{aligned}
$$

Solving for λ in this equation gives

$$
\lambda = - \frac{\int_{\mathbf{R}^3} \nabla \cdot \left(\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|} |\nabla\psi| \right) \delta(\phi)\delta(\psi)|\nabla\psi| dx}{\int_{\mathbf{R}^3} |P_{\nabla\psi}\nabla\phi| \delta(\phi)\delta(\psi)|\nabla\psi| dx}.
$$

All this together defines the evolution equation for ϕ that moves a curve by geodesic curvature flow while keeping the enclosed surface area fixed. This equation is also valid and can be used in higher dimensions. For more on the process of fixing enclosed area or volume, see [16].

Numerically, the right hand side of the evolution equation is handled in the manner corresponding to the flow. The left hand side is in Hamilton Jacobi form and we solve it as in the constant normal flow section, i.e., using third order TVD-RK in time and Hamilton Jacobi fifth order WENO-LLF in space. At each Runge-Kutta step, we solve for λ by using second order approximations for the integrals and using the ϕ from the previous step in the integrands.

To derive the evolution equation from the corresponding $\mathbf{R}^2$ equation,

$$
\phi_t + \lambda|\nabla\phi| = \nabla \cdot \nabla\beta(\nabla\phi)|\nabla\phi|,
$$

with

$$
\lambda = - \frac{\int_{\mathbf{R}^2} \nabla \cdot \nabla\beta(\nabla\phi)|\nabla\phi|\delta(\phi) dx}{\int_{\mathbf{R}^2} |\nabla\phi|\delta(\phi) dx},
$$

we note that all the terms have been considered previously except the λ term. For λ, we would like to take the integrals over the surface instead of over $\mathbf{R}^2$. This means we change the integral from $\int_{\mathbf{R}^2} dx$ to $\int_{\mathbf{R}^3} \delta(\psi)|\nabla\psi| dx$. Using this, the rest of the terms carry over as before and so writting the $\mathbf{R}^2$ equation on the surface gives the correct evolution equation.

We can also consider flows that are not minimizations of energies. If we want the curve to move according to the equation

$$\phi_t = -v \cdot \nabla\phi,$$

where v can depend on ψ, ϕ, and their derivatives, then the constrained motion can be given by

$$\phi_t + \lambda|P_{\nabla\psi}\nabla\phi| = -v \cdot \nabla\phi,$$

where

$$\lambda = -\frac{\int_{\mathbf{R}^3}(v \cdot \nabla\phi)\delta(\phi)\delta(\psi)|\nabla\psi|dx}{\int_{\mathbf{R}^3}|P_{\nabla\psi}\nabla\phi|\delta(\phi)\delta(\psi)|\nabla\psi|dx}.$$

This will move a curve according to v while keeping the enclosed surface area fixed. Note that this makes no mention of the evolution equation coming from minimizing an energy. However, when there is an energy for the flow, such as in geodesic curvature flow or Wulff flow, the evolution equation makes more sense. Also higher dimensions can be considered using the same evolution equations.

In Figure 3.15, we show a curve moving by geodesic curvature flow with a fixed enclosed surface area constraint on a paraboloid. The steady state curve is a circle symmetrically wrapped around the paraboloid. In Figure 3.16, we show a curve moving by the same flow on a sphere. The initial curve is elliptical in nature. The steady state curve is a circle on the sphere. We see that our method can easily handle the fixed enclosed surface area constraint for curves on surfaces.

3.11.1 Wulff Shapes

We can get Wulff shapes on a surface by running the evolution equation for Wulff flow with fixed enclosed surface area to get the steady state of the zero level set of ϕ on M. Following the steps for deriving the evolution equation for such a motion, we start with the energy

$$E(\phi) = \int_{\mathbf{R}^3} \beta\left(\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|}\right)|P_{\nabla\psi}\nabla\phi||\nabla\psi|\delta(\psi)\delta(\phi)dx - \lambda\int_{\mathbf{R}^3} H(-\phi)\delta(\psi)|\nabla\psi|dx,$$

where λ is a Lagrange multiplier, and get the evolution equation

$$\phi_t + \lambda|P_{\nabla\psi}\nabla\phi| = \nabla\cdot(P_{\nabla\psi}D\beta(P_{\nabla\psi}\nabla\phi)|\nabla\phi|)\frac{|P_{\nabla\psi}\nabla\phi|}{|\nabla\psi|},$$

where

$$\lambda = \frac{\int_{\mathbf{R}^3}\nabla\cdot(P_{\nabla\psi}D\beta(P_{\nabla\psi}\nabla\phi)|\nabla\phi|)\delta(\phi)\delta(\psi)|\nabla\psi|dx}{\int_{\mathbf{R}^3}|P_{\nabla\psi}\nabla\phi|\delta(\phi)\delta(\psi)|\nabla\psi|dx}.$$

The steady state curve on M of this evolution equation gives a Wulff shape on the surface. Wulff shapes in higher dimensions can be found using the same equations.

In Figure 3.17, we show a curve moving under Wulff flow while fixing the enclosed surface area. The final shape is a Wulff shape on the surface and is squarish in nature since we take $\beta(x)$ to be a smoothed out version of $|x_1| + |x_2| + |x_3|$. The algorithm can be used to find Wulff shapes on other surfaces as well.

3.12 Moving Curves on Moving Surfaces

We now extend our results to include moving curves on moving surfaces. Since the surface is moving, ψ will now depend on time and the zero level set of ψ at any time gives the surface at that time. Also, the curve on the surface at any time is given by the intersection of the zero level sets of ψ and ϕ at that time. To follow the surface and curve, we need only follow ψ and ϕ or, more accurately, the zero level set of ψ and the intersection of this with the zero level set of ϕ. The initial surface and curve are given and can be represented by an initial ψ and ϕ.

Now suppose the motion we want for the curve satisfies fixed surfaces, i.e., ψ fixed in time,

$$\phi_t + v \cdot \nabla\phi = 0,$$

for some velocity field v tangent to the level set surfaces of ψ that can depend on ψ, ϕ, and their derivatives. Suppose the motion of the surface itself satisfies

$$\psi_t + w \cdot \nabla\psi = 0,$$

for some velocity field w that can depend on ψ and its derivatives but not on ϕ in any way. Then we can get velocity field under which to move the curve by adding the two velocity fields v and w. This means the evolution equation is

$$\phi_t + (v + w) \cdot \nabla\phi = 0.$$

Note the curve and also the surface may undergo merging in the process of this evolution.

As an example, suppose we want the curve to move outward in its normal direction at unit speed. Then we get $v \cdot \nabla\phi = |P_{\nabla\psi}\nabla\phi|$. Suppose we also want the surface to move outward in its normal direction at unit speed. The equation for this is

$$\psi_t + |\nabla\psi| = 0,$$

with $w = \dfrac{\nabla\psi}{|\nabla\psi|}$. So $w \cdot \nabla\phi = \dfrac{\nabla\psi \cdot \nabla\phi}{|\nabla\psi|}$. The velocity fields together give the evolution equation for the desired motion of the curve,

$$\phi_t + P_{\nabla\psi}\nabla\phi + \frac{\nabla\psi \cdot \nabla\phi}{|\nabla\psi|} = 0.$$

Another example is where the curve itself does not move but the surface moves under the velocity field w. So the motion of the curve in $\mathbf{R}^3$ is due only to the motion of the surface. This specific problem, called region tracking, was first solved in [3] in using the same representation we use here. In this case,

$$\psi_t + w \cdot \nabla\psi = 0,$$

is the equation for the motion of the surface, and thus,

$$\phi_t + w \cdot \nabla\phi = 0,$$

is the equation for the motion of the curve.

All this can also be done for other previously described motions except for the case of fixed enclosed surface area. In this case, we need to clarify what we want since the surface may shrink until its total surface area is smaller than the enclosed surface area we want fixed. Higher dimensions are also covered by the above evolution equations.

A drawback of this method is that spurious curves may appear when surfaces merge. This happens when a part of the surface with negative values of ϕ touches a part of the surface with positive values. At the place of contact, a zero level set of ϕ is created in between the positive and negative values and so gives rise to a spurious curve on the surface. If the problem we are considering is curves on surfaces then this is a wrong answer. But if we change the problem, then the spurious curve actually makes sense. Let us think of the curve as the boundary of the set of negative values of ϕ on the surface and the movement of the curve as being due to the expansion or contraction of that set. The negative values may denote one substance and the positive values a different substance, as in two phase flow. When negative and positive values touch, a new boundary for the set of negative values needs to be created and, hence, we should get a curve appearing there to separate the positive and negative values. This way of thinking is not only convenient here but may also be useful in applications.

In Figure 3.18, we show initially a circle on a plane. The circle is moving by constant normal flow on the plane and the plane is moving by constant normal flow in $\mathbf{R}^3$. The final picture shows a dilated circle on a translated plane. More complicated curves and surfaces can also be handled by the algorithm.

3.13 Local Level Set Method

A curve is a one dimensional object, so to solve an evolution equation in all of $\mathbf{R}^3$ to move the curve is too expensive. In most cases, we only need to solve the equation in a neighborhood of the curve. Counterexamples, however, include getting signed distance functions to curves, where the ϕ is needed over the whole surface, and cases where curves can appear out of nowhere, for example, in the active contour method of Chan and Vese[5]. But in most of the motions we have studied here, the evolution equation is only needed in a neighborhood of the curve.

We have succeeded in localizing near the surface, i.e., retaining only the grid points that are near the surface. This is optimal for the methods that need ϕ defined on the whole surface. We create a data structure to hold only the grid points close to the surface. The structure only needs to be created once, at the beginning, since the surface is static. This immediately cuts down on our memory storage. Also, we solve our partial differential equations only at the retained grid points in this structure, thus greatly speeding up the method. To determine which grid points are near the surface and thus should be in the structure, we look at the distances in $\mathbf{R}^3$ of those points away from the surface. Only points under a certain value, a constant times Δx, are retained, which makes this method optimal when ϕ needs to be solved over the whole surface. We use the fast marching method once at the beginning to create the distance values at grid points.

In actuality, we will only solve our partial differential equation in a smaller neighborhood of the surface than the neighborhood of retained points. This is so the stencils of the finite difference schemes we use will not exit the neighborhood of retained points. Fortunately, the fast marching method we used to obtain distance to the surface, as a by-product, also gives an ordering of the points with respect to their distance values, from least to greatest. We can then use this to enforce Neumann boundary conditions on the boundary of the smaller neighborhood by extending the values there, following the normal vectors of the boundary, to the larger neighborhood. Note the normal vectors of the boundary are in the same direction as the gradients of the distance values and thus following the ordering given by the fast marching method correctly propagates the values. So even though the partial differential equation is only solved in the smaller neighborhood, the finite difference schemes will use values throughout the whole neighborhood. The method, up to this point, also retains accuracy.

We may also want to make sure that the boundary conditions will not affect the behavior of ϕ on the surface. For this, we can make the level set surfaces of ϕ perpandicular to the surface while not changing the values of ϕ on the surface. This is accomplished using the evolution equation described previously in Section 3.8 for this purpose. The fast marching method can also be used instead. The process of making the level sets of ϕ perpendicular to the surface, however, may reduce the accuracy of the method.

So far, we have constructed a local level set method that is optimal for solving partial differential equations over the whole surface but not for solving just in a neighborhood of a curve. For this, we currently have a method that has the potential to be optimal in both speed and memory but has not yet been programmed up in such a way. It similarly involves retaining only the grid points that are near the zero level sets of ψ and ϕ, solving in a smaller neighborhood of these retained points, and making the level sets on the surface well behaved as described previously in Section 3.8. This ensures that the boundaries of the neighborhood do not affect the motion of the curve. However, the data structure is no longer static since the curve is moving, taking the neighborhood along with it. This aspect slightly complicates the problem.

Table 3.5 shows the local level set method applied to constant normal flow. The evolution equations are solved over the whole surface. Also, the level sets of ϕ are not enforced to be perpandicular to the surface and so the accuracy is roughly second order (compare to Table 3.1). Running the algorithm to make the level sets of ϕ perpandicular to the surface will slightly move the contours on the surface and reduce the accuracy to first order. The table was generated by looking at a circle moving by constant normal flow on a sphere. Note, much finer grids can be used than for the global method. Table 3.6 shows the local level set method applied to finding signed distance functions. Once again, the level sets of ϕ are not enforced to be perpandicular to the surface. The result is first order accuracy, as in the global case. The table was generated by looking at a circle on a sphere. Note, much finer grids can be used than for the global method. All in all, the local level set method is faster and needs less memory than the global method while still preserving the accuracy of the method.

3.14 Higher Dimensions and Codimensions

We can further extend our method to higher codimensions by using more functions $\phi_1, \ldots, \phi_k$ and $\psi_1, \ldots, \psi_m$ in $\mathbf{R}^n$, for $k+m \leq n$. The intersection of the zero level sets of $\psi_1, \ldots, \psi_m$ gives the constraint, and the intersection of this with the

grid size	error	order
$8 \times 8 \times 8$	0.053125	
$16 \times 16 \times 16$	0.016274	1.7068
$32 \times 32 \times 32$	0.00561706	1.5347
$64 \times 64 \times 64$	0.00173415	1.6956
$128 \times 128 \times 128$	0.000433395	2.0005
$256 \times 256 \times 256$	0.000169642	1.3532
$320 \times 320 \times 320$	0.000107546	2.0425

Table 3.5: This is the order of accuracy analysis for the local level set method for constant normal flow. The example considered was a circle moving on a sphere. Because of the behavior of the error for the $256 \times 256 \times 256$ case, we say the method is roughly second order accurate. Note we can run on a $320 \times 320 \times 320$ grid with this algorithm.

intersection of the zero level sets of $\phi_1, \ldots, \phi_k$ gives the object to be moved under the constraint. This means the constraint surface has dimension $n - m$ and on this, we move an object having dimension $n - m - k$. The actual motions will be carried out under a system of evolution equations for $\phi_1, \ldots, \phi_k$. Note, however, the fact that our methods are grid based, usually using uniform grids, means the size of computer memory needed to run simulations in very high dimensions may be restrictive, even when using a local level set method.

3.15 Conclusion

We have devised a level set based method for moving curves constrained on surfaces. This method can accurately handle a wide variety of curves and surfaces

grid size	error	order
$10 \times 10 \times 10$	0.04269	
$20 \times 20 \times 20$	0.0300296	0.5075
$40 \times 40 \times 40$	0.0170282	0.8185
$80 \times 80 \times 80$	0.00864061	0.9787
$160 \times 160 \times 160$	0.00435087	0.9898
$320 \times 320 \times 320$	0.00213815	1.0249

Table 3.6: This is the order a accuracy analysis for the local level set method for distance functions. The curve considered was a circle on a sphere. The method is first order accurate. Note we can run on a $320 \times 320 \times 320$ grid with this algorithm.

and motions. It also extends all the results of the plane surface level set method and so it concievably has a wide range of applications. Basic applications already allow us to get signed distance functions, geodesics, and various interesting crystal shapes on surfaces. The limitations are just the limitations of any level set based approach. Finally, the method is easy to implement because complex surface topologies and procedures, such as merging or breaking or keeping the curve on the surface, are all handled automatically.

3.16 Proofs of Propositions

Proof of Proposition 3.1

We will prove that these identities hold in $\mathbf{R}^n$ for arbitrary n.

(a) This follows from the fact that P_w is a symmetric matrix and $P_w^2 = P_w$.

(b) This follow from the fact that

$$(P_X\nabla)_i u = P_X \nabla u \cdot e_i = \nabla u \cdot P_X e_i.$$

(c) We prove this property by brute force calculations and, for simplification, summing over repeated indices. Let e_i be the vector with 1 for its ith component and 0 for the rest. This means for the jth component, $(e_i)_j = \delta_{ij}$. So we have,

$$\begin{aligned}
\nabla^{\nabla u} \cdot (P_{\nabla u} X) &= \nabla_i^{\nabla u} (P_{\nabla u} X)_i \\
&= \nabla((P_{\nabla u} X)_i) \cdot P_{\nabla u} e_i \\
&= \left[\left(\delta_{ij} - \frac{u_{x_i} u_{x_j}}{|\nabla u|^2}\right) X_j\right]_{x_k} \left(\delta_{ki} - \frac{u_{x_k} u_{x_i}}{|\nabla u|^2}\right) \\
&= \left[\left(\delta_{ij} - \frac{u_{x_i} u_{x_j}}{|\nabla u|^2}\right) X_j\right]_{x_i} - \left[\left(\delta_{ij} - \frac{u_{x_i} u_{x_j}}{|\nabla u|^2}\right) X_j\right]_{x_k} \frac{u_{x_k} u_{x_i}}{|\nabla u|^2}.
\end{aligned}$$

Calling the first term I and the second term J, we have

$$I = \left[\left(\delta_{ij} - \frac{u_{x_i} u_{x_j}}{|\nabla u|^2}\right) X_j\right]_{x_i} = \nabla \cdot (P_{\nabla u} X),$$

and

$$\begin{aligned}
J &= -\left[\left(\delta_{ij} - \frac{u_{x_i} u_{x_j}}{|\nabla u|^2}\right) X_j\right]_{x_k} \frac{u_{x_k} u_{x_i}}{|\nabla u|^2} \\
&= -\frac{u_{x_k} u_{x_i} (X_i)_{x_k}}{|\nabla u|^2} + \left(\frac{u_{x_i} u_{x_j} X_j}{|\nabla u|^2}\right)_{x_k} \frac{u_{x_k} u_{x_i}}{|\nabla u|^2} \\
&= -\frac{u_{x_k} u_{x_i} (X_i)_{x_k}}{|\nabla u|^2} + \frac{u_{x_i} u_{x_j} (X_j)_{x_k} u_{x_k} u_{x_i}}{|\nabla u|^4} + \left(\frac{u_{x_i} u_{x_j}}{|\nabla u|^2}\right)_{x_k} \frac{X_j u_{x_k} u_{x_i}}{|\nabla u|^2} \\
&= \left(\frac{u_{x_j} u_{x_i}}{|\nabla u|^2}\right)_{x_k} \frac{X_j u_{x_k} u_{x_i}}{|\nabla u|^2} \\
&= \left(\frac{u_{x_i} u_{x_j x_k}}{|\nabla u|^2} + \frac{u_{x_j} u_{x_i x_k}}{|\nabla u|^2} - \frac{2 u_{x_j} u_{x_i} u_{x_m} u_{x_m x_k}}{|\nabla u|^4}\right) \frac{X_j u_{x_k} u_{x_i}}{|\nabla u|^2} \\
&= \frac{u_{x_k} u_{x_j x_k} X_j}{|\nabla u|^2} - \frac{u_{x_i} u_{x_j} u_{x_k} u_{x_i x_k} X_j}{|\nabla u|^4} \\
&= \left(X_i - \frac{u_{x_i} u_{x_j} X_j}{|\nabla u|^2}\right) \frac{u_{x_k} u_{x_i x_k}}{|\nabla u|^2} \\
&= P_{\nabla u} X \cdot \frac{\nabla |\nabla u|}{|\nabla u|}.
\end{aligned}$$

So altogether,

$$\nabla^{\nabla u}\cdot(P_{\nabla u}X)=\nabla\cdot(P_{\nabla u}X|\nabla u|)\frac{1}{|\nabla u|},$$

which completes the proof.

Proof of Proposition 3.2 We will prove the results in general for $\mathbf{R}^n$. We sum over repeated indices for convenience.

(a) Fix a point x on M. Let ν be the outward unit normal vector to M at x. Now given two orthonormal bases in $\mathbf{R}^n$, e_i and $\tilde{e}_i$, $i=1,\ldots,n$, let δ_i, let δ_i and $\tilde{\delta}_i$ be $(P_\nu\nabla)_i$ under the frames e_i and $\tilde{e}_i$, respectively. This means

$$\begin{aligned}\delta_i &= \left(\delta_{ij}-\frac{\langle\nu,e_i\rangle\langle\nu,e_j\rangle}{|\nu|^2}\right)\partial_j\\ \tilde{\delta}_i &= \left(\delta_{ij}-\frac{\langle\nu,\tilde{e}_i\rangle\langle\nu,\tilde{e}_j\rangle}{|\nu|^2}\right)\tilde{\partial}_j,\end{aligned}$$

where ∂_i and $\tilde{\partial}_i$ correspond to the frames e_i and $\tilde{e}_i$, respectively. Because of orthonormality, we have that $\tilde{e}_i=a_{ij}e_j$ with the a_{ij} forming an orthogonal matrix, i.e., $a_{ij}a_{ik}=a_{ji}a_{ki}=\delta_{jk}$. Thus, we have $\tilde{\delta}_i=a_{ij}\delta_j$. Therefore,

$$\tilde{e}_i\tilde{\delta}_i=e_i\delta_i.$$

Now taking e_i, $i=1,\ldots,n$, to be the standard orthonormal basis in $\mathbf{R}^n$ and $\tilde{e}_i$, $i=1,\ldots,n$, to be an orthonormal basis of $\mathbf{R}^n$ with $\tilde{e}_n=\nu$, we get

$$\begin{aligned}\tilde{e}_i\tilde{\delta}_i &= \nabla^S\\ e_i\delta_i &= P_\nu\nabla.\end{aligned}$$

So

$$\nabla^S=P_\nu\nabla,$$

and, especially, if u is a real valued function in $\mathbf{R}^n$, then

$$\nabla^S u=P_\nu\nabla u.$$

(b) Continuing with the above computations, given X a vector field in $\mathbf{R}^n$, we have

$$\langle \tilde{\delta}_i X, \tilde{e}_i \rangle = \langle \delta_i X, e_i \rangle.$$

Therefore,

$$\nabla^S \cdot X = P_\nu \nabla X.$$

Proof of Proposition 3.3 Apply Proposition 3.7 with $\beta(p) = |p|$, $p \in \mathbf{R}^3$.

Proof of Proposition 3.4 Let e_1, e_2, e_3 be orthonormal. Then

$$\nabla \cdot X = \langle e_1(X), e_1 \rangle + \langle e_2(X), e_2 \rangle + \langle e_3(X), e_3 \rangle,$$

where $e_1(f_1, f_2, f_3) = (e_1(f_1), e_2(f_2), e_3(f_3))$. This is since for $\tilde{e}_1, \tilde{e}_2, \tilde{e}_3$ orthonormal, $\tilde{e}_i = \sum_{j=1}^3 a_i^j e_j$, so

$$\begin{aligned}
\sum_{i=1}^3 \langle \tilde{e}_i(X), \tilde{e}_i \rangle &= \sum_{i=1}^3 \langle \sum_{j=1}^3 a_i^j e_j(X), \sum_{k=1}^3 a_i^k e_k \rangle \\
&= \sum_{i=1}^3 \sum_{j=1}^3 \sum_{k=1}^3 a_i^j a_i^k \langle e_j(X), e_k \rangle \\
&= \sum_{j=1}^3 \sum_{k=1}^3 \delta_{jk} \langle e_j(X), e_k \rangle \\
&= \sum_{j=1}^3 \langle e_j(X), e_j \rangle.
\end{aligned}$$

Therefore, if $\tilde{e}_1 = (1,0,0), \tilde{e}_2 = (0,1,0), \tilde{e}_3 = (0,0,1)$, we get

$$\sum_{i=1}^3 \langle \tilde{e}_i(X), \tilde{e}_i \rangle = \nabla \cdot X = \sum_{j=1}^3 \langle e_j(X), e_j \rangle.$$

Now, let $e_1 = \dfrac{\nabla\psi \times \nabla\phi}{|\nabla\psi \times \nabla\phi|}, e_2 = \dfrac{e_1 \times \nabla\psi}{|e_1 \times \nabla\psi|}, e_3 = e_1 \times e_2$. Note e_1, e_2, e_3 is orthonormal and the quantity we wish to investigate is $\nabla \cdot (e_1 \times \nabla\psi)$. Now

$$\nabla \cdot (e_1 \times \nabla\psi) = \langle e_1(e_1 \times \nabla\psi), e_1 \rangle + \langle e_2(e_1 \times \nabla\psi), e_2 \rangle + \langle e_3(e_1 \times \nabla\psi), e_3 \rangle.$$

But

$$e_1(e_1 \times \nabla\psi) = e_1(e_1) \times \nabla\psi + e_1 \times (e_1(\nabla\psi)),$$

so

$$\begin{aligned}\langle e_1(e_1 \times \nabla\psi), e_1\rangle &= \langle e_1(e_1) \times \nabla\psi, e_1\rangle \\ &= \langle \kappa \times \nabla\psi, e_1\rangle \\ &= \det(\kappa, \nabla\psi, e_1) \\ &= \kappa \cdot (\nabla\psi \times e_1).\end{aligned}$$

Also,

$$\langle e_2(e_1 \times \nabla\psi), e_2\rangle = \langle e_2(e_1) \times \nabla\psi, e_2\rangle + \langle e_1 \times (e_2(\nabla\psi)), e_2\rangle.$$

Now e_1 is a unit vector field implies $e_2(e_1) \perp e_1$, which means $e_2(e_1)$ is a linear combination of $\nabla\psi$ and e_2. Therefore,

$$\langle e_2(e_1) \times \nabla\psi, e_2\rangle = 0,$$

and so

$$\langle e_2(e_1 \times \nabla\psi), e_2\rangle = \langle e_1 \times (e_2(\nabla\psi)), e_2\rangle.$$

Finally,

$$\langle e_3(e_1 \times \nabla\psi), e_3\rangle = \langle e_3(e_1) \times \nabla\psi, e_3\rangle + \langle e_1 \times (e_3(\nabla\psi)), e_3\rangle.$$

Now e_1 is a unit vector field implies $e_3(e_1) \perp e_1$, which means $e_3(e_1)$ is a linear combination of $\nabla\psi$ and e_2. Since

$$\langle e_2 \times \nabla\psi\rangle = -\langle e_2 \times e_3, \nabla\psi\rangle = -\langle e_1, \nabla\psi\rangle = 0,$$

therefore,

$$\langle e_3(e_1 \times \nabla\psi), e_3\rangle = \langle e_1 \times (e_3(\nabla\psi)), e_3\rangle.$$

So altogether, we have

$$\nabla \cdot (e_1 \times \nabla\psi) = \kappa \cdot (\nabla\psi \times e_1) + \langle e_1 \times (e_2(\nabla\psi)), e_2\rangle + \langle e_1 \times (e_3(\nabla\psi)), e_3\rangle.$$

But

$$\begin{aligned} \langle e_1 \times (e_2(\nabla\psi)), e_2\rangle &= -\langle e_1 \times e_2, e_2(\nabla\psi)\rangle \\ &= -\langle e_3, e_2(\nabla\psi)\rangle, \end{aligned}$$

and

$$\begin{aligned} \langle e_1 \times (e_3(\nabla\psi)), e_3\rangle &= \langle e_3 \times e_1, e_3(\nabla\psi)\rangle \\ &= \langle e_2, e_3(\nabla\psi)\rangle. \end{aligned}$$

Now

$$\langle e_2, e_3(\nabla\psi)\rangle = \sum_{i=1}^{3}\sum_{j=1}^{3} a_i b_j \frac{\partial^2\psi}{\partial x_i \partial x_j} = \sum_{i=1}^{3}\sum_{j=1}^{3} b_i a_j \frac{\partial^2\psi}{\partial x_i \partial x_j} = \langle e_3, e_2(\nabla\psi)\rangle.$$

Therefore,

$$\nabla \cdot (e_1 \times \nabla\psi) = \kappa \cdot (\nabla\psi \times e_1).$$

Proof of Proposition 3.5 See main body of section for proof.

Proof of Proposition 3.6 With $\beta(p) = |p|$, we get $D\beta(p) = \frac{p}{|p|}$ and $D^2\beta(p) = \frac{1}{|p|}P_p$. So

$$D^2\beta\left(\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|}\right) = P_{P_{\nabla\psi}\nabla\phi}.$$

Therefore, using Proposition 3.8, we get that the principle matrix for the right hand side of the evolution equation is $P_{\nabla\psi}P_{P_{\nabla\psi}\nabla\phi}P_{\nabla\psi}$ and also since $P_{P_{\nabla\psi}\nabla\phi}$ is nonnegative definite, with one zero eigenvalue and the rest being equal to one, the evolution equation is thus degenerate parabolic.

To actually find the eigenvalues of the principle matrix, we note that $\nabla\psi$ and $P_{\nabla\psi}\nabla\phi$ are eigenvectors corresponding to the zero eigenvalue, since $P_{\nabla\psi}\nabla\psi = 0$

and $P_{P_{\nabla\psi}\nabla\phi}P_{\nabla\psi}P_{\nabla\psi}\nabla\phi = P_{P_{\nabla\psi}\nabla\phi}P_{\nabla\psi}\nabla\phi = 0$, respectively. Also, given any other vector, $v \in \mathbf{R}^3$, perpendicular to these two eigenvectors, we have

$$P_{\nabla\psi}P_{P_{\nabla\psi}\nabla\phi}P_{\nabla\psi}v = P_{\nabla\psi}P_{P_{\nabla\psi}\nabla\phi}v = P_{\nabla\psi}v = v.$$

Therefore, we can conclude that the principle matrix has two zero eigenvalues with the rest being equal to one.

Proof of Proposition 3.7 We have, since β is a homogeneous of degree one function,

$$\begin{aligned} E(\phi) &= \int_{\mathbf{R}^3} \beta\left(\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|}\right)|P_{\nabla\psi}\nabla\phi||\nabla\psi|\delta(\psi)\delta(\phi)dx \\ &= \int_{\mathbf{R}^3} \beta(P_{\nabla\psi}\nabla\phi)\delta(\phi)|\nabla\psi|\delta(\psi). \end{aligned}$$

So,

$$\begin{aligned} \frac{d}{ds}\bigg|_{s=0} E(\phi+s\eta) &= \int_{\mathbf{R}^3}(D\beta(P_{\nabla\psi}\nabla\phi)\cdot P_{\nabla\psi}\nabla\eta)\delta(\phi)|\nabla\psi|\delta(\psi)dx + \\ &\quad + \int_{\mathbf{R}^3}\beta(P_{\nabla\psi}\nabla\phi)\delta'(\phi)|\nabla\psi|\delta(\psi)\eta dx \\ &= I + J, \end{aligned}$$

where I is the first integral on the right hand side and J is the second. Now,

$$\begin{aligned} I &= \int_{\mathbf{R}^3}\left(D\beta(P_{\nabla\psi}\nabla\phi)\cdot\left(\nabla\eta - \frac{\nabla\psi\cdot\nabla\eta}{|\nabla\psi|^2}\nabla\psi\right)\right)\delta(\phi)|\nabla\psi|\delta(\psi)dx \\ &= -\int_{\mathbf{R}^3}\nabla\cdot(D\beta(P_{\nabla\psi}\nabla\phi)\delta(\phi)|\nabla\psi|\delta(\psi))\eta dx + \\ &\quad + \int_{\mathbf{R}^3}\nabla\cdot\left((D\beta(P_{\nabla\psi}\nabla\phi)\cdot\nabla\psi)\nabla\psi\delta(\phi)\frac{\delta(\psi)}{|\nabla\psi|}\right)\eta dx \\ &= -\int_{\mathbf{R}^3}\nabla\cdot(D\beta(P_{\nabla\psi}\nabla\phi)|\nabla\psi|\delta(\psi))\delta(\phi)\eta dx + \\ &\quad - \int_{\mathbf{R}^3}(D\beta(P_{\nabla\psi}\nabla\phi)\cdot\nabla\phi)\delta'(\phi)|\nabla\psi|\delta(\psi)\eta dx + \\ &\quad + \int_{\mathbf{R}^3}\nabla\cdot\left((D\beta(P_{\nabla\psi}\nabla\phi)\cdot\nabla\psi)\nabla\psi\frac{\delta(\psi)}{|\nabla\psi|}\right)\delta(\phi)\eta dx + \end{aligned}$$

$$\begin{aligned}
&+\int_{\mathbf{R}^3}(D\beta(P_{\nabla\psi}\nabla\phi)\cdot\nabla\psi)(\nabla\psi\cdot\nabla\phi)\delta'(\phi)\frac{\delta(\psi)}{|\nabla\psi|}\eta dx\\
= &-\int_{\mathbf{R}^3}\nabla\cdot\Big(D\beta(P_{\nabla\psi}\nabla\phi)|\nabla\psi|\delta(\psi)-\\
&\qquad(D\beta(P_{\nabla\psi}\nabla\phi)\cdot\nabla\psi)\nabla\psi\frac{\delta(\psi)}{|\nabla\psi|}\Big)\,\delta(\phi)\eta dx+\\
&-\int_{\mathbf{R}^3}\Big((D\beta(P_{\nabla\psi}\nabla\phi)\cdot\nabla\phi)|\nabla\psi|-\\
&\qquad(D\beta(P_{\nabla\psi}\nabla\phi)\cdot\nabla\psi)\frac{\nabla\psi\cdot\nabla\phi}{|\nabla\psi|}\Big)\,\delta'(\phi)\delta(\psi)\eta dx\\
= &-\int_{\mathbf{R}^3}\nabla\cdot(P_{\nabla\psi}D\beta(P_{\nabla\psi}\nabla\phi)|\nabla\psi|\delta(\psi))\delta(\phi)\eta dx+\\
&-\int_{\mathbf{R}^3}(P_{\nabla\psi}\nabla\phi\cdot D\beta(P_{\nabla\psi}\nabla\phi))\delta'(\phi)|\nabla\psi|\delta(\psi)\eta dx.
\end{aligned}$$

Note,

$$\nabla\cdot(P_{\nabla\psi}D\beta(P_{\nabla\psi}\nabla\phi)|\nabla\psi|\delta(\psi))=\nabla\cdot(P_{\nabla\psi}D\beta(P_{\nabla\psi}\nabla\phi)|\nabla\psi|)\delta(\psi),$$

since $P_{\nabla\psi}D\beta(P_{\nabla\psi}\nabla\phi)\cdot\nabla\psi=0$.

Also for $p\in\mathbf{R}^3$, we have

$$D\left(\beta\left(\frac{p}{|p|}\right)\right)=\frac{1}{|p|}D\beta\left(\frac{p}{|p|}\right)-\frac{p}{|p|^3}\left(D\beta\left(\frac{p}{|p|}\right)\cdot p\right),$$

which means, since $\beta(p)=|p|\beta\left(\frac{p}{|p|}\right)$,

$$D\beta(p)=\frac{p}{|p|}\beta\left(\frac{p}{|p|}\right)+D\beta\left(\frac{p}{|p|}\right)-\frac{p}{|p|}\left(D\beta\left(\frac{p}{|p|}\right)\cdot\frac{p}{|p|}\right).$$

Therefore,

$$p\cdot D\beta(p)=|p|\beta\left(\frac{p}{|p|}\right)=\beta(p).$$

Note especially that $p=P_{\nabla\psi}\nabla\phi$ gives

$$P_{\nabla\psi}\nabla\phi\cdot D\beta(P_{\nabla\psi}\nabla\phi)=\beta(P_{\nabla\psi}\nabla\phi).$$

So altogether, we get

$$\left.\frac{d}{ds}\right|_{s=0}E(\phi+s\eta)=-\int_{\mathbf{R}^3}\nabla\cdot(P_{\nabla\psi}D\beta(P_{\nabla\psi}\nabla\phi)|\nabla\psi|)\delta(\psi)\delta(\phi)\eta dx,$$

and so $\frac{d}{ds}\Big|_{s=0} E(\phi + s\eta) = 0$ for arbitrary η leads to the Euler-Lagrange equation

$$-\nabla \cdot (P_{\nabla\psi} D\beta(P_{\nabla\psi}\nabla\phi)|\nabla\psi|)\delta(\psi)\delta(\phi) = 0.$$

Proof of Proposition 3.8 Let $F(q,x) = P_{\nabla\psi} D\beta(P_{\nabla\psi} q)|\nabla\psi|$. Then we have

$$(F_i)_{q_j}(q,x) = P_{\nabla\psi} D^2\beta(P_{\nabla\psi} q) P_{\nabla\psi}|\nabla\psi|,$$

where $D^2\beta$ is the Hessian matrix for β.

Therefore, the principle matrix for $\nabla \cdot (P_{\nabla\psi} D\beta(P_{\nabla\psi}\nabla\phi)|\nabla\psi|)\frac{|P_{\nabla\psi}|}{|\nabla\psi|}$ is

$$P_{\nabla\psi} D^2\beta(P_{\nabla\psi}\nabla\phi) P_{\nabla\psi}|P_{\nabla\psi}\nabla\phi|,$$

which can be rewritten as

$$P_{\nabla\psi} D^2\beta\left(\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|}\right) P_{\nabla\psi},$$

since $D^2\beta$ is homogeneous of degree -1.

Therefore, if the matrix $N = D^2\beta\left(\frac{P_{\nabla\psi}\nabla\phi}{|P_{\nabla\psi}\nabla\phi|}\right)$ is nonnegative definite, then $x^T N x \geq 0$ for all $x \in \mathbf{R}^3$. This implies for any $y \in \mathbf{R}^3$, taking $x = P_{\nabla\psi} y$ gives that $y^T P_{\nabla\psi} N P_{\nabla\psi} y \geq 0$ and so $P_{\nabla\psi} N P_{\nabla\psi}$ is also nonnegative definite. Note $\nabla\psi$ is an eigenvector corresponding to the 0 eigenvalue.

So we have shown if N is nonnegative definite, then the evolution equation

$$\phi_t = \nabla \cdot (P_{\nabla\psi} D\beta(P_{\nabla\psi}\nabla\phi)|\nabla\psi|)\frac{|P_{\nabla\psi}|}{|\nabla\psi|}$$

is degenerate parabolic.

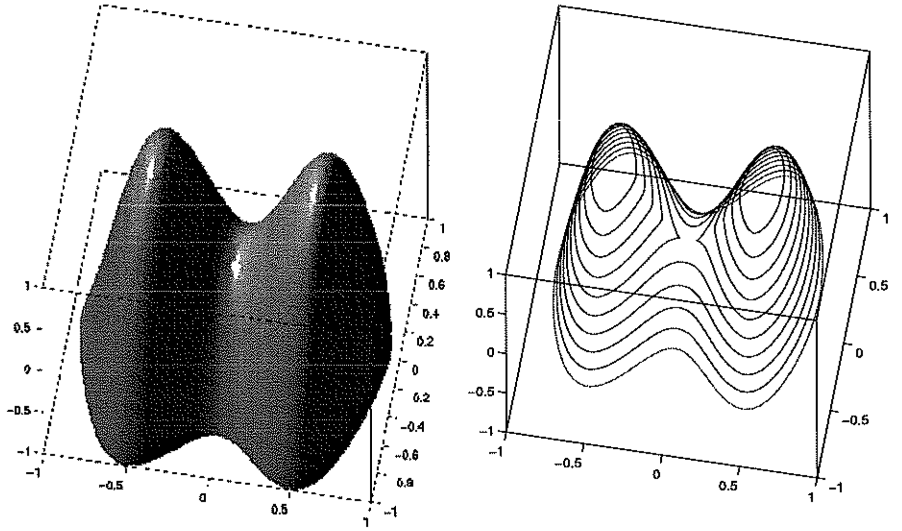

Figure 3.1: The surface, two mountains, is shown on the left and the curves are shown separately on the right. These are curves moving inward by unit normal flow. Note that the curve breaks into two pieces during the flow.

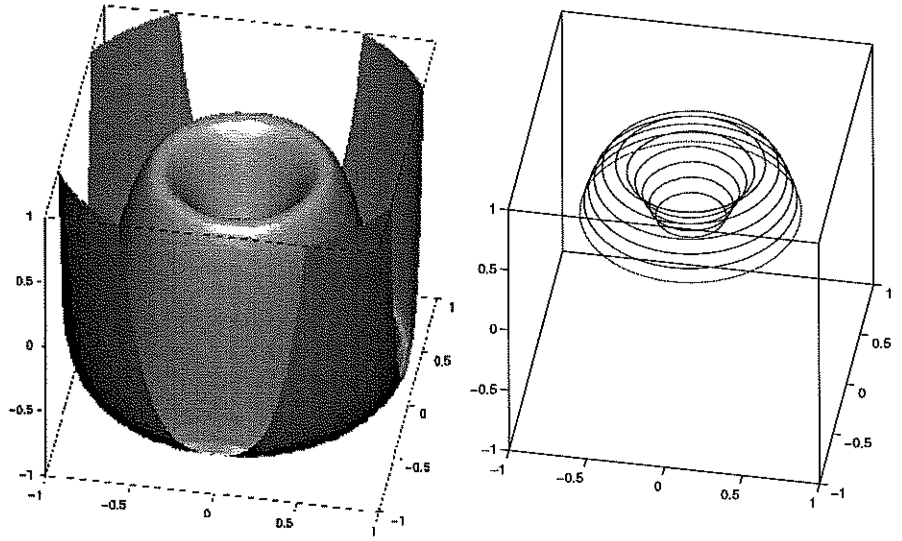

Figure 3.2: The surface, a volcano, is shown on the left and the curves are shown separately on the right. These are curves moving inward by unit normal flow and go up the side of the volcano and back down into it.

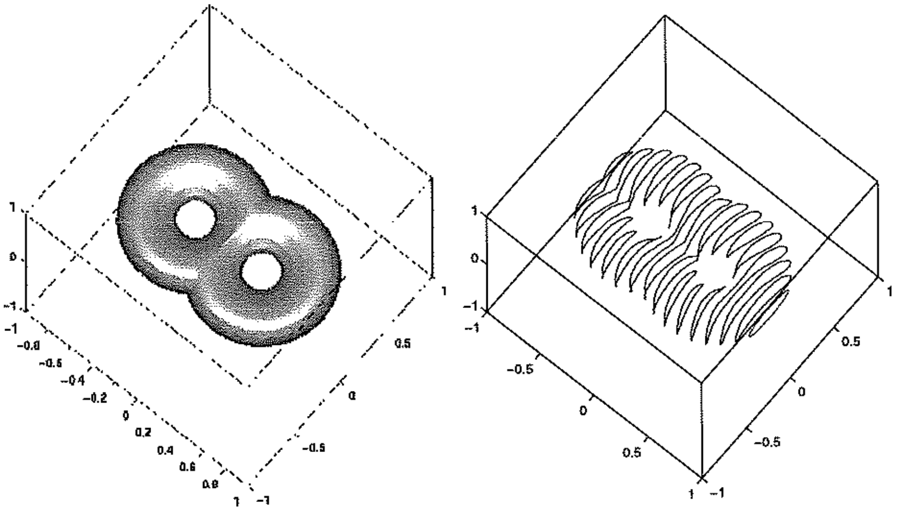

Figure 3.3: The surface is shown on the left and the curves are shown separately on the right. These are curves moving inward by unit normal flow. The curve translates to the left on the two holed torus, breaking into pieces and merging back again a few times.

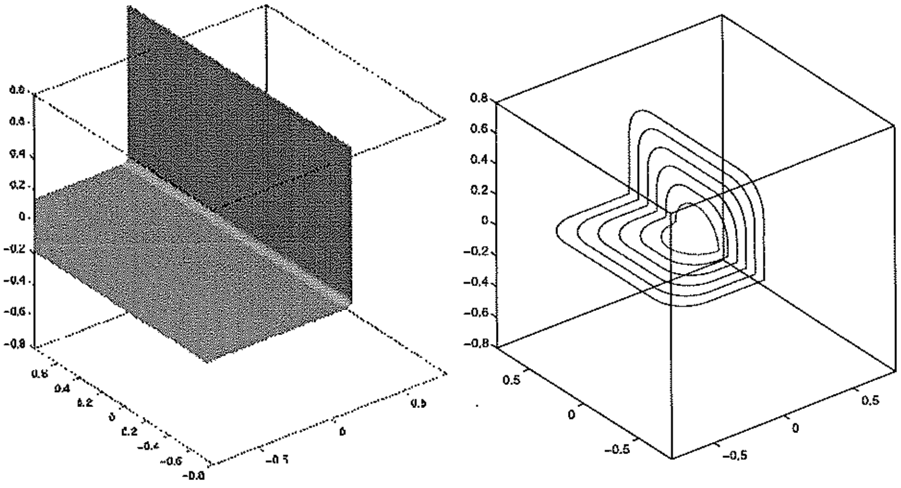

Figure 3.4: The surface is shown on the left and the curves are shown separately on the right. These are curves moving outward in the normal direction by a non-constant speed. The chosen speed causes the curve to develop a squarish aspect as it grows.

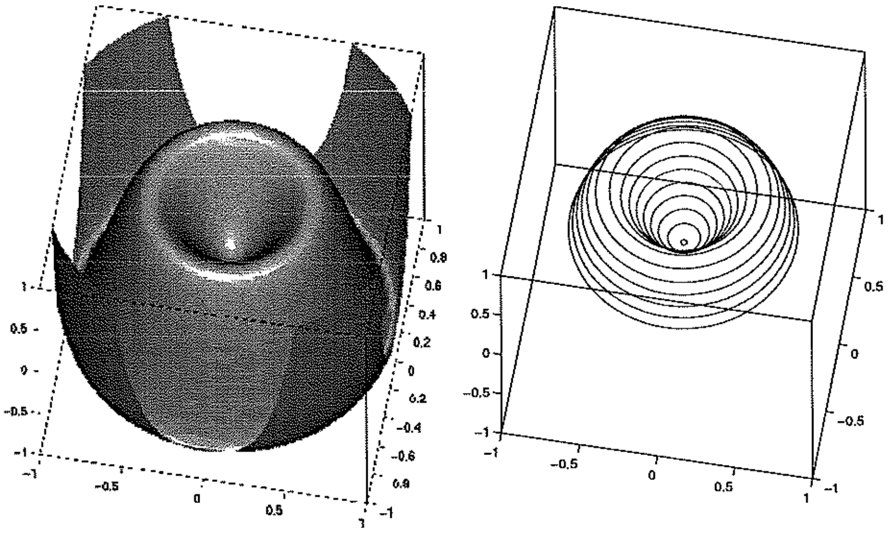

Figure 3.5: The surface is shown on the left and the contours of the distance function are shown on the right. The picture is similar to that of constant normal flow on a volcano.

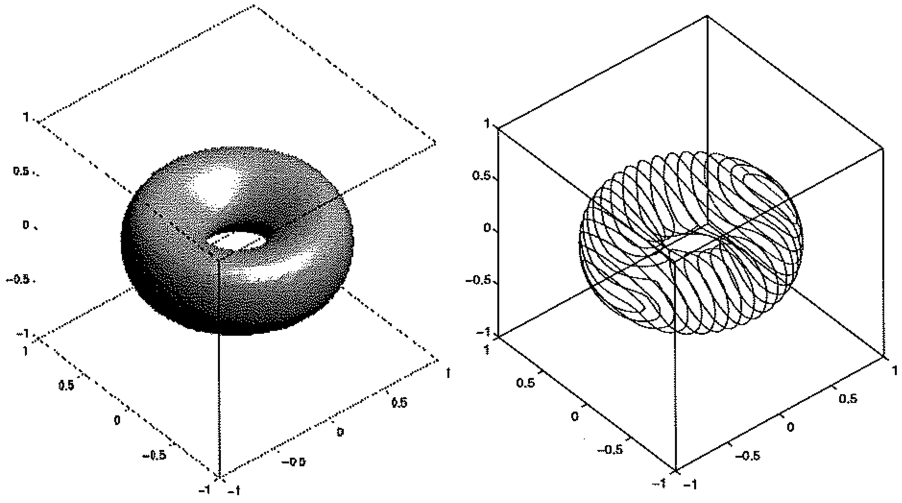

Figure 3.6: The surface is shown on the left and the contours of the distance function are shown on the right. The surface is a torus.

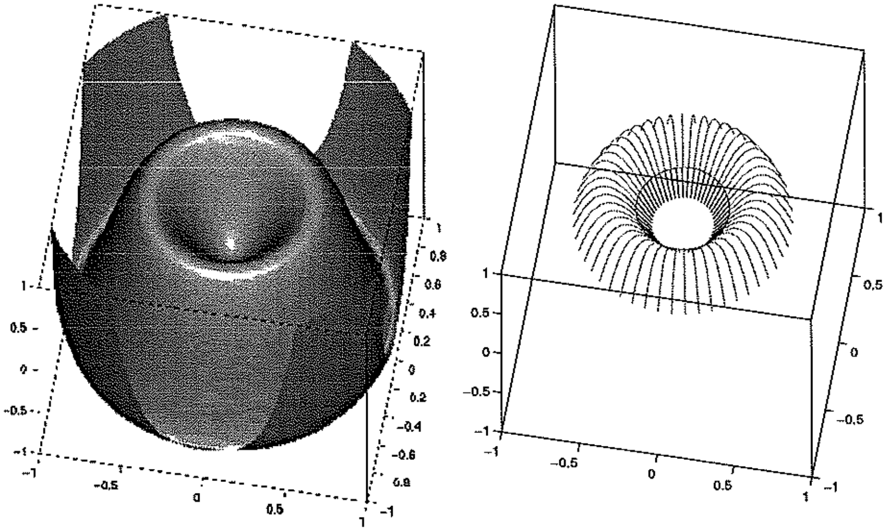

Figure 3.7: The volcano surface is shown on the left and the geodesics from various points to a curve in the volcano core are shown on the right.

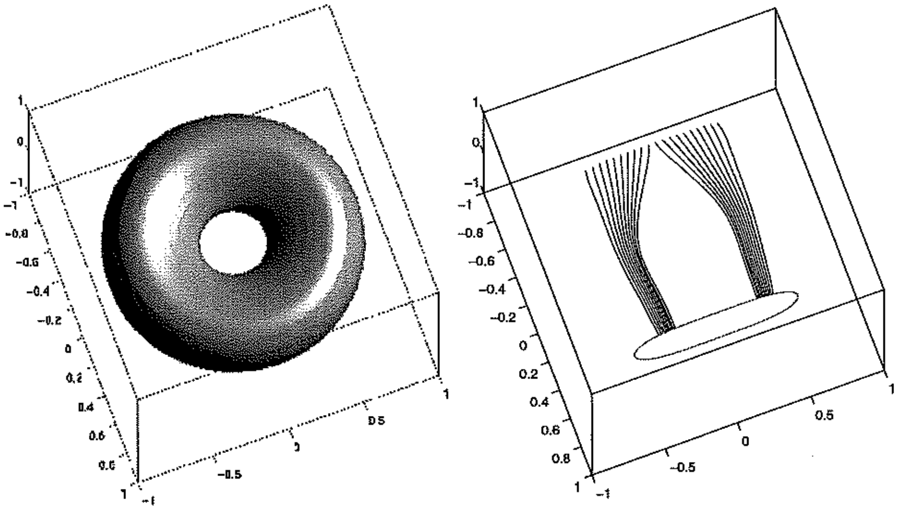

Figure 3.8: A torus is shown on the left and the geodesics from various points to a curve on the torus are shown on the right.

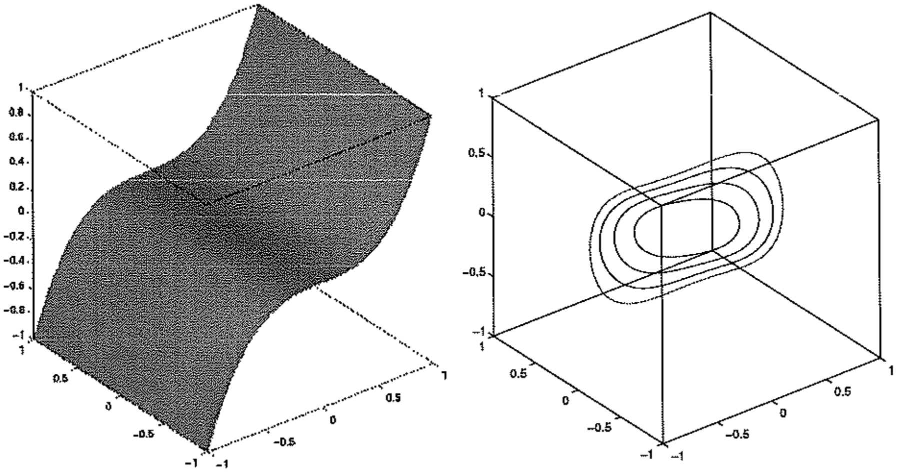

Figure 3.9: The surface is shown on the left and the evolution of the curve under geodesic curvature flow is shown on the right. The curve shrinks on the surface.

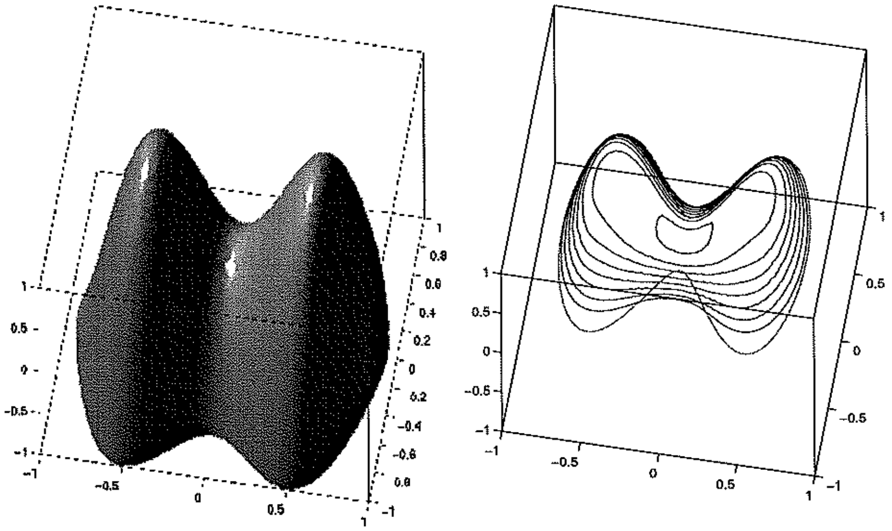

Figure 3.10: The surface, two mountains, is shown on the left and the evolution of the curve under geodesic curvature flow is shown on the right. The curve needs to move over the mountains before it can shrink to a point and disappear.

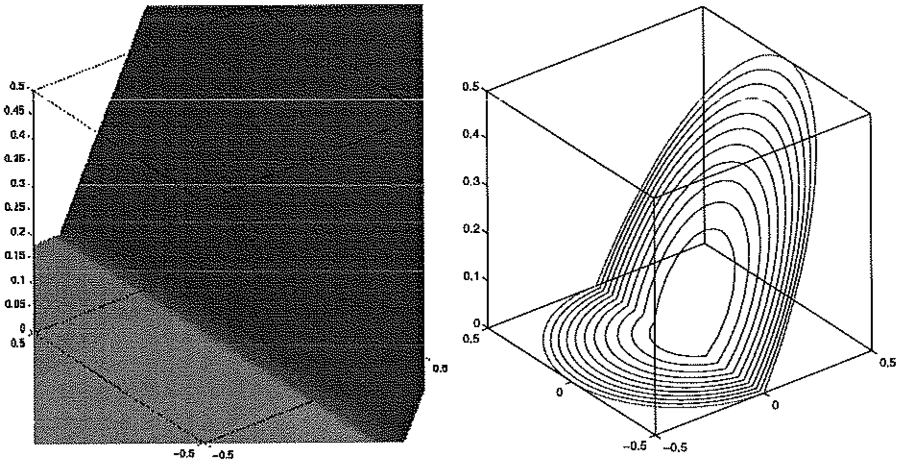

Figure 3.11: The surface, a bent plane, is shown on the left and the evolution of the curve under geodesic curvature flow is shown on the right. Note the surface has a kink in it.

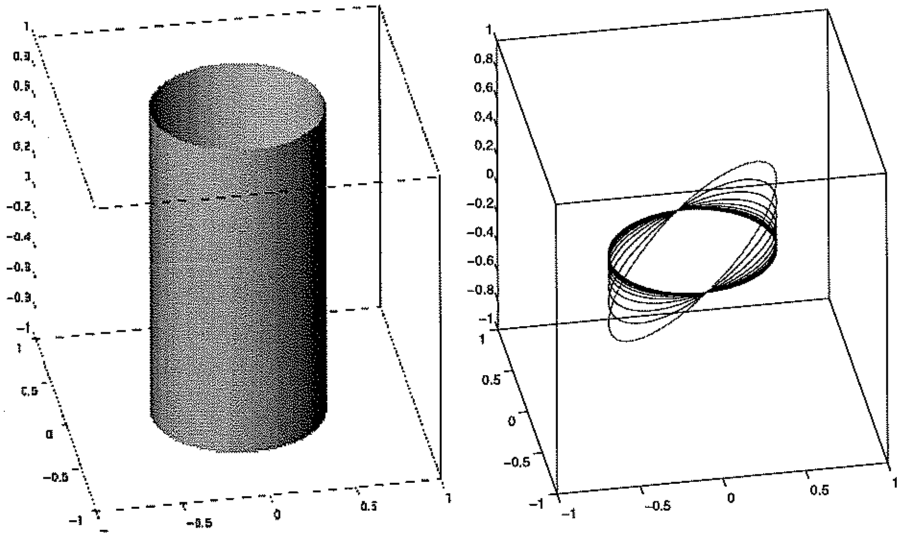

Figure 3.12: The surface, a cylinder, is shown on the left and the evolution of the curve under geodesic curvature flow is shown on the right. The curve ends up wrapping tightly around the cylinder, forming a geodesic curve on the surface.

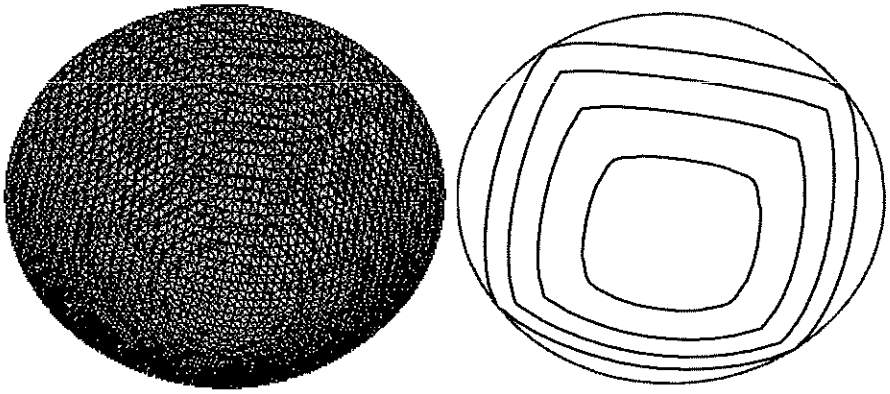

Figure 3.13: The surface, the bottom of a paraboloid, is shown on the left and the evolution of the curve under Wulff flow, with $\beta(x)$ a smoothed out form of $|x_1| + |x_2| + |x_3|$, is shown on the right. The curve develops a squarish shape before disappearing.

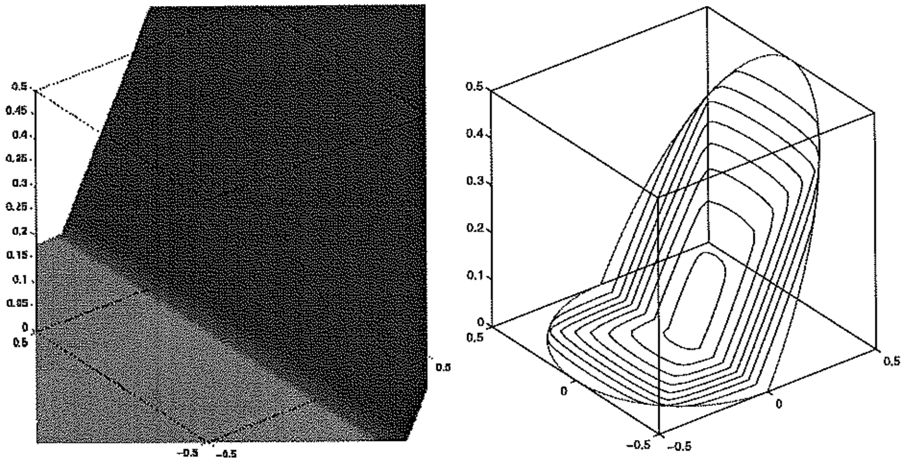

Figure 3.14: The surface, a bent plane, is shown on the left and the evolution of the curve under Wulff flow, with $\beta(x)$ a smoothed out form of $|x_1| + |x_2| + |x_3|$, is shown on the right. The curve develops a squarish shape on the surface.

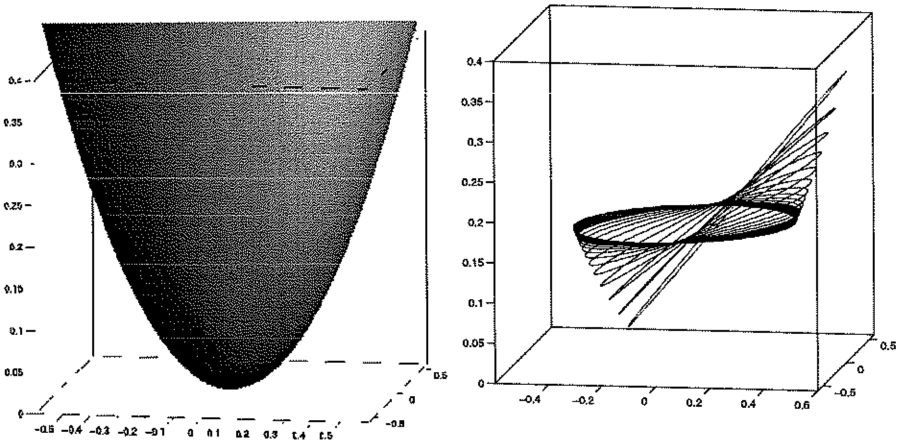

Figure 3.15: The surface, a paraboloid, is shown on the left and the evolution of the curve under geodesic curvature flow with fixed enclosed surface area is shown on the right. The initial curve evolves to a steady state curve, a circle symmetrically wrapped around the paraboloid.

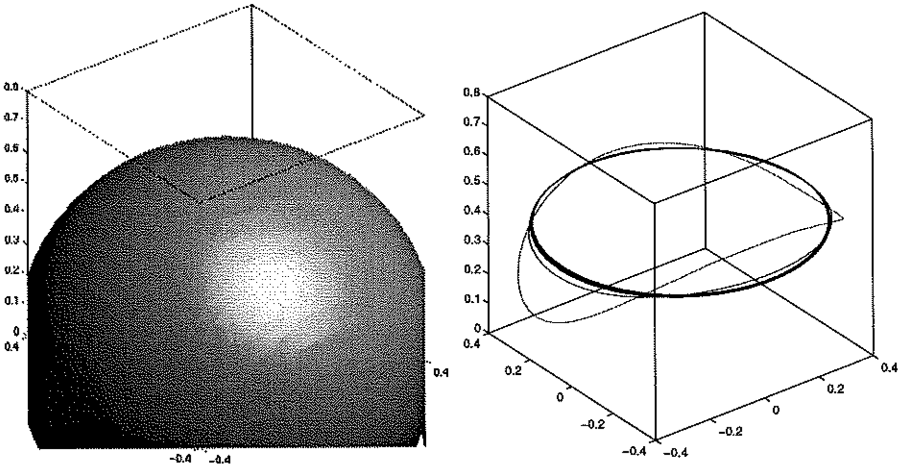

Figure 3.16: The surface, a sphere, is shown on the left and the evolution of the curve under geodesic curvature flow with fixed enclosed surface area is shown on the right. The initial curve evolves to a steady state curve, a circle on the sphere.

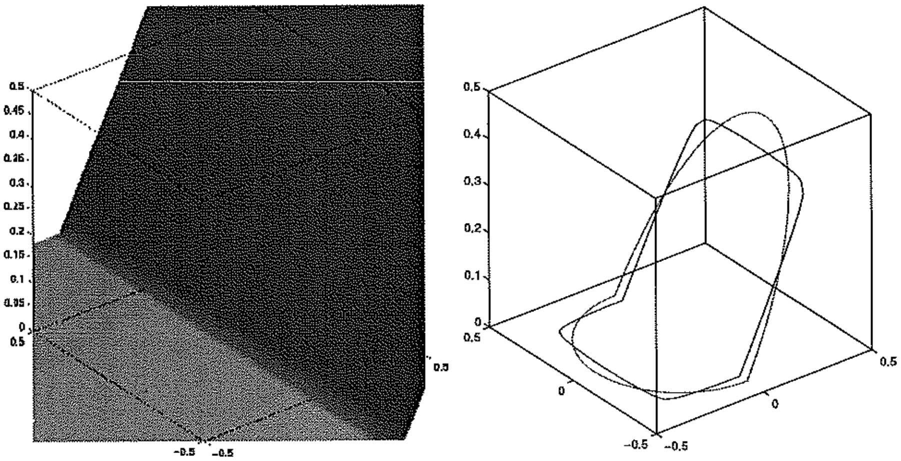

Figure 3.17: The surface, a bent plane, is shown on the left and the evolution of the curve under Wulff flow with fixed enclosed surface area is shown on the right. The initial curve evolves to a steady state curve, a smoothed out squarish shape.

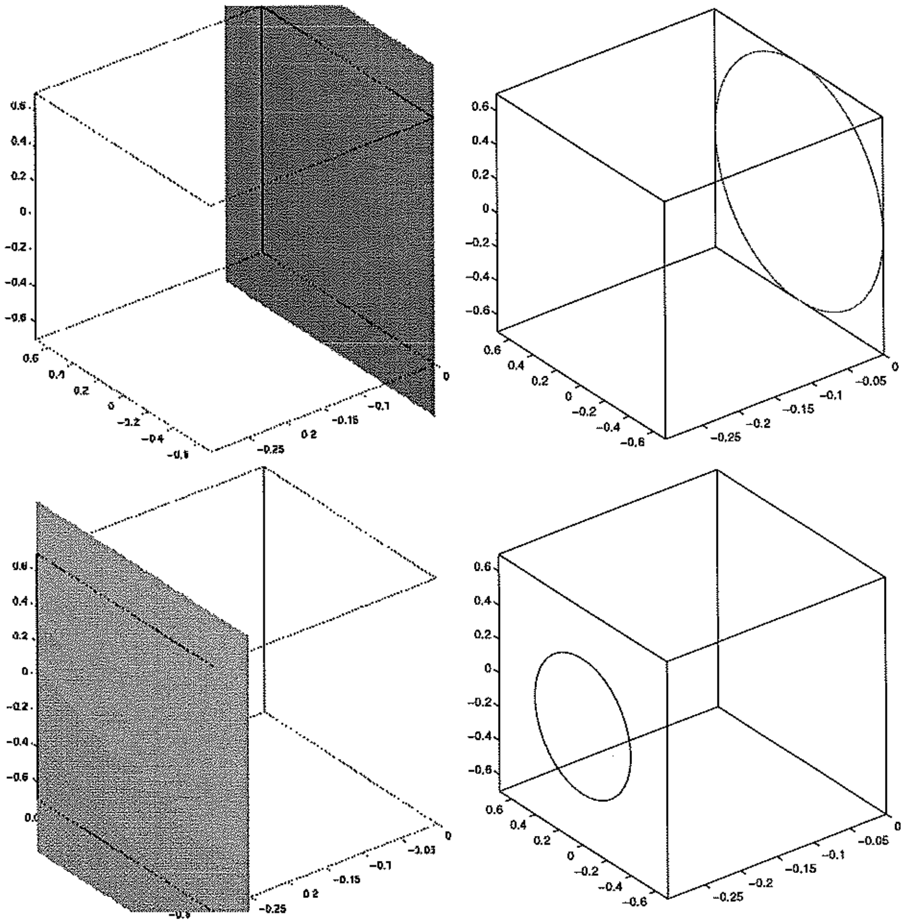

Figure 3.18: This is a moving curve on moving surface computation. The original surface and curve are shown in the two plots on the top. The final surface and curve are shown in the two plots below. The surface and curve are both moving by constant normal flow.

CHAPTER 4

A Level Set Based Method for Denoising and Deblurring Images on Surfaces

4.1 Abstract

We study a partial differential equation based method for the processing of images painted on surfaces. One can find images on surfaces in, for example, digital effects for motion pictures. It also has use in the medical regime on three dimensional scans of organs such as brain scans. The method we introduce here deals with the denoising and deblurring of grey scale images on hypersurfaces. Results will show that the same tools used in image processing for two dimensional images[26] can be extended for use on surfaces.

4.2 Setup

Our method is based on the representation in Chapter 3. Let M be the surface we are interested in and let v be a function on M that denotes the grey scale values of the image on the surface. Then we take a level set function ψ over $\mathbf{R}^3$ such that its zero level set is M and we take a function u over $\mathbf{R}^3$ such that the restriction of u to M is v. ψ is considered to be static in time since the surface is not moving. The evolution of u in time, on the other hand, will evolve the image on M, keeping in mind that u restricted to M is the image we are interested in. Thus we want to create partial differential equations for u whose solutions are either the denoised or deblurred image on the surface.

4.3 The Evolution Equations for Denoising and Deblurring

We wish to create the evolution equations for denoising and deblurring on M.

4.3.1 Denoising

To create the partial differential equations that will do what we want, we take a hint from image processing of two dimensional images. Let v_0 be a two dimensional grey scale image, i.e., v_0 is a function defined on $\mathbf{R}^2$ whose values are the grey scale values of the image. For denoising, we start with v_0 a noisy image. Then solving to steady state the equation

$$v_t = \nabla \cdot \frac{\nabla v}{|\nabla v|} - \lambda(v - v0),$$

gives a good denoised solution. This is because the equation comes from gradient descent on minimizing the energy

$$E(v) = \int_{\mathbf{R}^2} |\nabla v| dx + \frac{\lambda}{2} \int_{\mathbf{R}^2} (v - v_0)^2 dx.$$

The first term is the total variation of the image. The second term keeps the denoised image close to the original noisy image. So altogether, the above equation minimizes the total variation of the image while keeping it close to the original image. Note λ is a parameter that weighs the importance of the terms and must be chosen wisely. Minimizing the total variation, instead of something else, is very important because this will not only squash the noise, but will preserve sharp edges without Gibbs phenomenon. Previous attempts have used

$$\int_{\mathbf{R}^2} |\nabla v|^2 dx,$$

instead of the first term. This smears out edges and may even add a ringing effect near them. Total variation thus seems to be the best choice.

We can then use the tools in Chapter 3 to derive the same equation for functions on surfaces. The quick way involves putting the partial differential equation directly onto the surface. The equation for u in $\mathbf{R}^3$ then becomes

$$u_t = \nabla \cdot \left(\frac{P_{\nabla\psi} \nabla u}{|P_{\nabla\psi} \nabla u|} |\nabla \psi| \right) \frac{1}{|\nabla \psi|} - \lambda(u - u_0),$$

where u_0 restricted to M is the initial noisy image on the surface and $P_{\nabla\psi}$ orthogonally projects vectors onto the plane with normal vector $\nabla\psi$.

The slower but more intuitive way involves rewritting the energy as an energy on the surface. Thus E becomes

$$E(u) = \int_{\mathbf{R}^3} |\nabla^S u| \delta(\psi) |\nabla \psi| dx + \frac{\lambda}{2} \int_{\mathbf{R}^3} (u - u_0)^2 \delta(\psi) |\nabla \psi| dx,$$

where ∇^S is the surface gradient vector operator. Notice we used the fact that

$$\int_M w dA = \int_{\mathbf{R}^3} w \delta(\psi) |\nabla \psi| dx,$$

for any function w in $\mathbf{R}^3$. We have the equality $\nabla^S u = P_{\nabla\psi} \nabla u$, and so the energy can be written as

$$E(u) = \int_{\mathbf{R}^3} |P_{\nabla\psi} \nabla u| \delta(\psi) |\nabla \psi| dx + \frac{\lambda}{2} \int_{\mathbf{R}^3} (u - u_0)^2 \delta(\psi) |\nabla \psi| dx.$$

The first term is still the total variation except now for functions on a surface. The second term still keeps the solution close to the original given image on the surface. We may then find the Euler-Lagrange equation for this energy and perform modified gradient descent on it. This will give exactly the same equation we got in the quick way, noting that $\delta(\psi)$ should be replaced by $\frac{1}{|\nabla\psi|}$. Also, the partial differential equation in fact denoises each image u on each level set surface of ψ.

Thus we have derived a partial differential equation for use on images painted on surfaces that preserves the properties of its counterpart for two dimensional images. As in Chapter 3, the use of ψ allows us to consider surfaces of all kinds of topologies, for example, two holed torii. No coordinate patches are used and the surface does not have to be triangulated.

4.3.2 Deblurring

We will derive the partial differential equation for deblurring in the same way. As before, we first look at the deblurring equation for two dimensional images. Let v_0 be the initial blurred image. Assume the blurring of an image v comes from

$$J * v = \int_{\mathbf{R}^2} J(x,y)v(y)dy,$$

where J is a function in $\mathbf{R}^2$. Thus if J is the delta function, then the image is not blurred at all and if J is the Gaussian kernel, then the image gets a Gaussian blur. So the deblurring equation takes the form

$$v_t = \nabla \cdot \frac{\nabla v}{|\nabla v|} - \lambda J^T * (J * v - v0),$$

coming from gradient descent minimizing the energy

$$E(v) = \int_{\mathbf{R}^2} |\nabla v| dx + \frac{\lambda}{2} \int_{\mathbf{R}^2} (J * v - v_0)^2 dx.$$

Here, we are using the notation $J^T(x,y) = J(y,x)$. Note the energy still has as its first term total variation. The second term now keeps the blur of the solution close to the original blurred image. As before, λ is a parameter weighing the importance of the two terms and should be chosen wisely.

Putting this partial differential equation onto the surface gives the equation

$$u_t = \nabla \cdot \left(\frac{P_{\nabla\psi}\nabla u}{|P_{\nabla\psi}\nabla u|} |\nabla\psi| \right) \frac{1}{|\nabla\psi|} - \lambda J^T * (J * u - u_0),$$

where

$$J * u = \int_{\mathbf{R}^3} J(x,y) u(y) \delta(\psi) |\nabla\psi| dy.$$

blurs the image u on M. For the values of J off the zero level set, we can require that

$$J * u = \int_{\mathbf{R}^3} J(x,y) u(y) \delta(\psi - C) |\nabla\psi| dy.$$

blurs u on the C level set surface of ψ in the same way. This gives the deblurring partial differential equation for images on surfaces.

The equation can also be derived from performing gradient descent minimizing the correct energy on the surface. This energy, coming from its counterpart for two dimensional images, takes the form

$$E(u) = \int_{\mathbf{R}^3} |\nabla^S u| \delta(\psi) |\nabla\psi| dx + \frac{\lambda}{2} \int_{\mathbf{R}^3} (J * u - u_0)^2 \delta(\psi) |\nabla\psi| dx,$$

which can be rewritten as

$$E(u) = \int_{\mathbf{R}^3} |P_{\nabla\psi}\nabla u| \delta(\psi) |\nabla\psi| dx + \frac{\lambda}{2} \int_{\mathbf{R}^3} (J * u - u_0)^2 \delta(\psi) |\nabla\psi| dx.$$

Note that the first term is still total variation on the surface. So, all in all, we are still minimizing total variation to get sharp edges while we deblur. Note the partial differential equation in fact deblurs each image u on each level set surface of ψ.

4.4 Discretization of the Equations

We lay down a uniform grid over $\mathbf{R}^3$. This simplifies finite difference scheme construction and implementation. For both the denoising and deblurring equations for images on surfaces, we use second order central differencing on all the spacial derivatives. We also use third order TVD-RK on the time derivative. λ is chosen by trial and error to get the best one. Getting ψ and u from M and v is also relatively easy. There are tools for extending values off an interface to all of $\mathbf{R}^3$ (see [6]).

For deblurring, we usually use for J the Gaussian kernel on surfaces. This means instead of computing $J * u$ using integrals, we can compute it by iterating the heat equation for the surface a few steps. This equation, which can be derived from its two dimensional counterpart or from the energy that it minimizes, takes the form

$$u_t = \nabla \cdot (P_{\nabla\psi} \nabla u |\nabla \psi|) \frac{1}{|\nabla \psi|}.$$

As for the other equations, all spacial derivatives are discretized using second order central differencing. For the time derivative, we can use Forward Euler.

The CFL conditions for the denoising and deblurring equation on surfaces say that the time step Δt needs to be less than a constant times Δx^3, where Δx is the spacial step. This restriction is the same as when dealing with two dimensional images. For the heat equation on surfaces, the CFL condition is slightly better and it says that Δt needs to be less than a constant times Δx^2. Also, it may be possible to relax the CFL conditions for our evolution equations as is done in [21].

We have programmed up these equations in three dimensions, i.e., for grey scale images on surfaces, and in two dimensions, i.e., for grey scale images on curves. For deblurring, we have so far only studied deblurring in the absence of noise. Results show that edges are indeed preserved and so the denoise and deblurred images are of good quality. We have also so far used only rather coarse grids and compute in all of $\mathbf{R}^3$. But with the local method described in Chapter

3, we will be able to handle much finer grids and obtain even better results at faster speeds while using the less memory.

Figures 4.1 shows a step function with added noise over a circle in $\mathbf{R}^2$. The final result using our scheme is a denoised function with sharp jumps. Figure 4.2 shows a heat blurred and noisy step function over a circle in $\mathbf{R}^2$. The final result deblurs and denoises while retaining sharp jumps.

In Figure 4.3, we show the original image, a white band over a black background on a cylinder, and the noisy image with white noise. Figure 4.4 shows the denoised image. Notice that the edge between the black and white regions is sharp. The triangular artifacts at the edge come from the method used for plotting an image on a surface, not from the denoising algorithm. Dealing with such edges is a total variation denoising algorithm's specialty. A rather coarse grid of 50 by 50 by 50 grid points was used.

In Figure 4.5, we show the original image, a picture of lena wrapped around a cylinder, and the noisy image with white noise. Figure 4.6 then shows the denoised image. A grid of 50 by 50 by 50 grid points was used.

In Figure 4.7, we show the original image, a picture of lena wrapped around a cylinder, and the blurred image with Gaussian blur. Figure 4.8 then shows the deblurred image. A grid of 50 by 50 by 50 grid points was used.

4.5 Conclusion

We have introduced a level set method for denoising and deblurring images painted on surfaces. The partial differential equations involved in part minimize the total variation of the image and hence keep edges sharp. These equations have given excellent results for two dimensional images and the results for images on surfaces are also good, even on coarse grids. This method thus can be very useful in the area of digital effects or three dimensional surface scans.

Figures 4.1 shows a step function with added noise over a circle in $\mathbf{R}^2$. The final result using our scheme is a denoised function with sharp jumps. Figure 4.2 shows a blurred and noisy step function over a circle in $\mathbf{R}^2$. The final result deblurs and denoises while retaining sharp jumps.

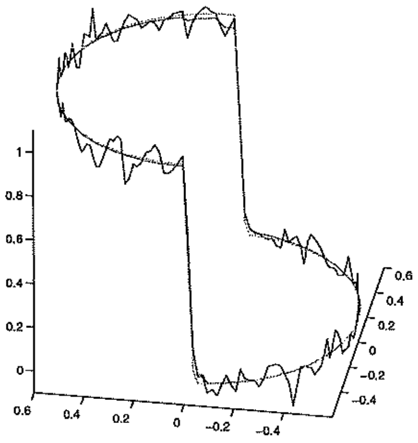

Figure 4.1: The original function is a step function over a circle. Noise is added and the denoised image retains sharp jumps.

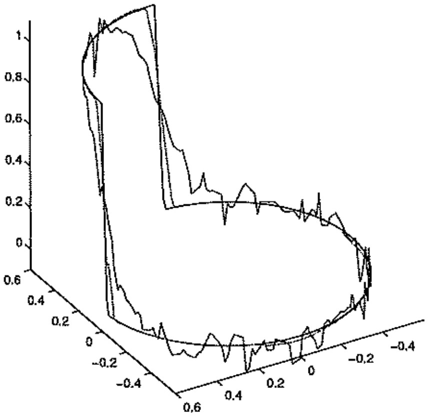

Figure 4.2: The original function is a step function over a circle. Heat blur is added along with noise and the deblurred and denoised image retains sharp jumps.

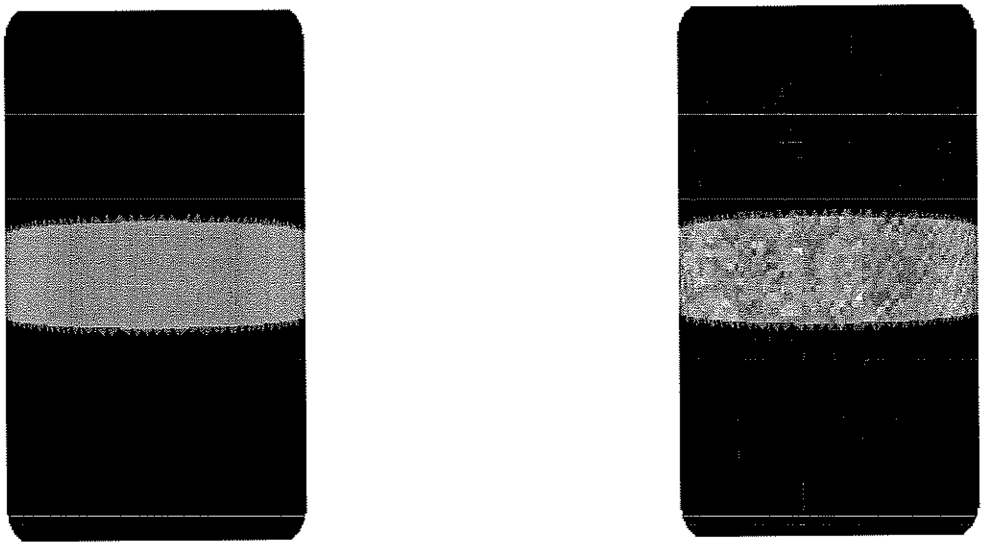

Figure 4.3: This is the original image (left) on a cylinder of a white band on a black background, and the noisy image (right) with white noise.

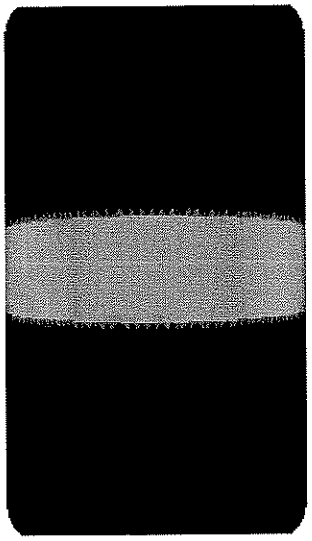

Figure 4.4: This is the denoised image using our TV denoising algorithm on surfaces. Notice sharp edges are recovered. Any triangular artifacts come from the plotter.

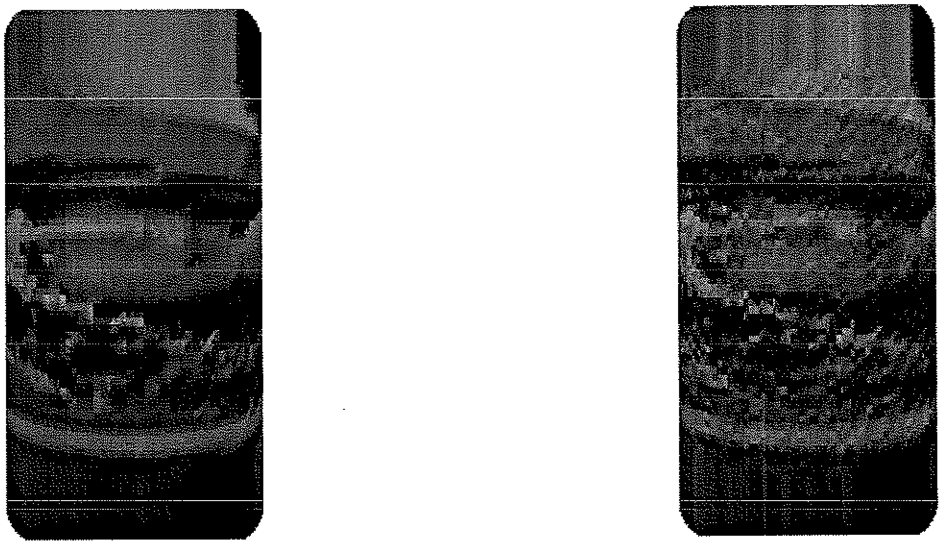

Figure 4.5: This is the original image (left) on a cylinder of a picture of lena wrapped around a cylinder, and the noisy image (right) with white noise.

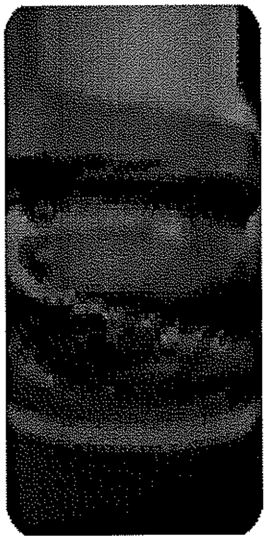

Figure 4.6: This is the denoised image using our TV denoising algorithm on surfaces. Any triangular artifacts come from the plotter.

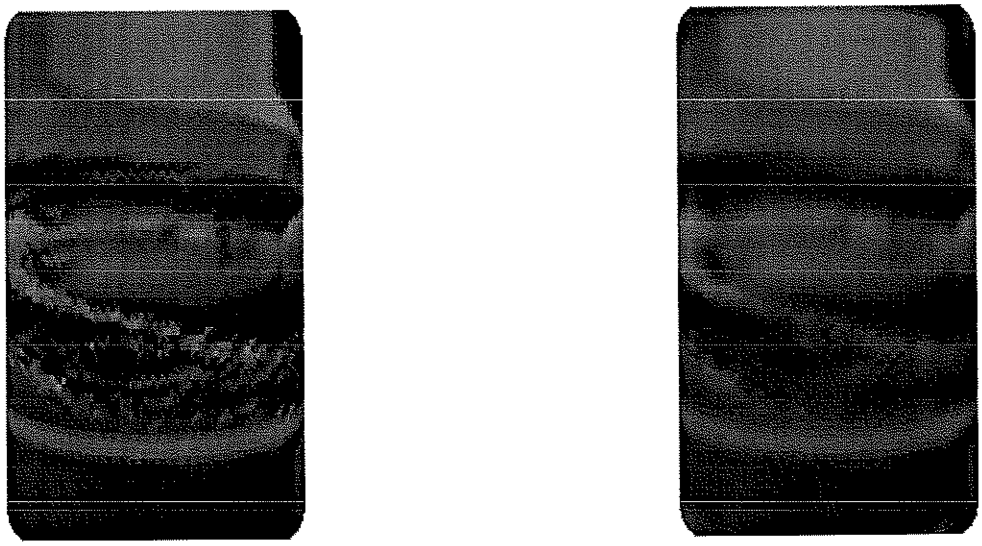

Figure 4.7: This is the original image (left) on a cylinder of a picture of lena wrapped around a cylinder, and the blurred image (right) with Gaussian blur.

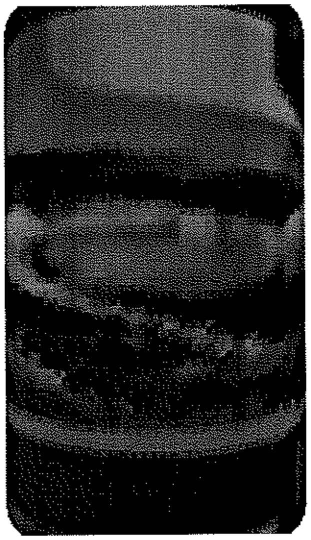

Figure 4.8: This is the deblurred image using our TV deblurring algorithm on surfaces. Any triangular artifacts come from the plotter.

CHAPTER 5

A Level Set Method Applied to the Minkowski Problem

5.1 Abstract

We use a level set approach to construct a partial differential equation for generating solutions to the Minkowski Problem. In this method, an initial compact, strictly convex hypersurface, represented by the zero level set of a function, flows to a final shape that is a dilation of the true solution. We will derive the general form of the partial differential equation using standard tools in level set theory and numerically compute solutions subject to different prescribed Gauss-Kronecker curvatures. Results will show the level set method applied to this problem can easily generate numerical solutions to the Minkowski Problem.

5.2 Setup

Given a compact, strictly convex hypersurface M in $\mathbf{R}^{n+1}$, one can define a Gauss map $G\colon M \to S^n$ such that $G(x) \in S^n$ is parallel to and in the direction of the unit inward normal at x. The map G is a diffeomorphism of M into S^n. Therefore, any data on M can be transplanted into data on S^n. Let K be the Gauss-Kronecker curvature on M. Then $K \circ G^{-1}$ is a function on S^n and satisfies

$$G_*(d\mu_M) = \frac{1}{K \circ G^{-1}} d\mu_{S^n},$$

where $d\mu_M$ and $d\mu_{S^n}$ denote the Lebesgue measures on M and S^n, respectively. Note that if x^i denotes the i^{th} coordinate function on S^n, then $x^i \circ G$ is a function on M. In fact, for $p \in M$, $x^i \circ G(p)$ denotes the i^{th} coordinate of the unit normal at p. Therefore,

$$\int_M (x^i \circ G) d\mu_M = 0 = \int_{S^n} x^i \frac{1}{K \circ G^{-1}} d\mu_{S^n}$$

shows that there are some compatibility conditions for a positive function on S^n to satisfy in order for it to be the Gauss-Kronecker curvature of a certain ovaloid. The Minkowski Problem says that this compatibility condition is sufficient.

Minkowski Problem. Given $F > 0$ on S^n such that for $i = 1, ..., n+1$,

$$\int_{S^n} x^i \frac{1}{F} d\mu_{S^n} = 0,$$

there exists a compact, strictly convex hypersurface M such that if $G: M \to S^n$ is the Gauss map, then the Gauss-Kronecker curvature at $p \in M$ equals to $F(G(p))$. Moreover, if M' is another solution, then M and M' differ by a translation.

This problem was solved in general dimensions by Cheng and Yau[7].

On the other hand, we are trying to generate solutions to the Minkowski Problem. This means given a function F satisfying the hypotheses of the Minkowski Problem, we wish to construct the corresponding hypersurface M. Our method uses the zero level set of a real valued function in $\mathbf{R}^{n+1}$, called a level set function, to represent a surface. So the evolution of the level set function gives the motion of the hypersurface. For definiteness, we also require a level set function ψ to be negative inside the surface and positive outside so that the inward pointing normal is given by $-\frac{\nabla\psi}{|\nabla\psi|}$. We then call such a function a solution to the Minkowski

Problem if and only if its zero level set is M. Note these solutions are not unique but their zero level set surfaces are. Also, we will mainly study the case $n = 2$ since this is the case that can be easily visualized. We will periodically comment on the validity of our formulas in other dimensions.

Other methods proposed include one by Chou and Wang[10]. This method is proven to converge starting with specific initial shapes but may diverge if the shapes are slightly perturbed.

5.3 Constructing The Partial Differential Equation

We wish to create a partial differential equation whose steady state solution is a solution to the Minkowski Problem. Thus we first attempt to find a function H such that the level set function ψ is a solution to the Minkowski Problem if and only if

$$H(\nabla^2\psi, \nabla\psi, \psi) = 0.$$

Then a partial differential equation can be constructed,

$$\phi_t = H(\nabla^2\phi, \nabla\phi, \phi)|\nabla\phi|,$$

with the initial condition

$$\phi(t = 0) = \phi_0,$$

where ϕ depends on x and t and ϕ_0 has compact, strictly convex level set surfaces.

The extra $|\nabla\phi|$ term on the right hand side of the partial differential equation is standard in level set methods. Its effect is to move the level sets of ϕ in the normal direction. In this case, the level sets of ϕ will move with speed $-H(\nabla^2\phi, \nabla\phi, \phi)$ in the outward normal direction. In general, given a vector field v, the equation

$$\phi_t + v \cdot \nabla\phi = 0,$$

moves the level sets of ϕ according to v. So if the vector field takes the form $v = v_0 \frac{\nabla\phi}{|\nabla\phi|}$, then the equation moves the level sets of ϕ in the normal direction with speed v_0. Note if v is perpendicular to the normal vector, then $v \cdot \nabla\phi = 0$. So after decomposing a general v with respect to an orthonormal basis containing the normal vector, we see that $v \cdot \nabla\phi = v_0|\nabla\phi|$ for some v_0. So all motions of level sets are motions in the normal direction, which means the equation

$$\phi_t + v_0|\nabla\phi| = 0,$$

is the general equation for moving level sets of ϕ.

Some conditions need to be satisfied by the partial differential equation in order to effectively solve the Minkowski Problem. One condition is that the partial differential equation needs to have a level set function whose zero level set is the true solution as an asymptotically stable steady state solution in order to be useful. Otherwise it will be nearly impossible to flow to the true solution. Another important condition is that the partial differential equation cannot be backward parabolic. We mention these two because it turns out that many simple guesses for H will not satisfy one of these properties.

For simplification of notation, we denote

$$\begin{aligned} H &= H(\nabla^2\phi, \nabla\phi, \phi) \\ K &= K(\nabla^2\phi, \nabla\phi) \\ F &= F\left(-\tfrac{\nabla\phi}{|\nabla\phi|}\right). \end{aligned}$$

5.3.1 Simple Guesses for H

We can make the simple guess $H = K - F$ to get, for the steady state surface, $K = F$. However, the solution to the Minkowski Problem here is not a stable steady state solution. Note if the exact solution of the Minkowski problem is a ball of radius r and if we start with the initial surface a ball of radius smaller than r, then it will shrink as time progresses and thus evolve away from the true solution. This is because the shape of the surface at any time will still be a ball but with a different radius than the initial one. To find the radius of the ball, we note that the Gauss-Kronecker curvature K for the initial surface is larger than the prescribed curvature F at any point on the surface. This gives $H > 0$, a property that holds for all time. So the initial ball will shrink and will not converge to the true solution. Similarly if we start with a ball of radius greater than r then this ball will expand away from the true solution. So there is no convergence and the partial differential equation does not have the solution of the Minkowski Problem as a stable steady state solution. This means numerically we cannot get to the true solution with this H.

Note also, the guess $H = F - K$ will not work since it is backward elliptic. This is because there is a negative sign in front of K and so the partial differential equation will move surfaces outward by Gauss-Kronecker curvature, an unstable flow.

Other choices include $H = \frac{1}{K} - \frac{1}{F}$ and $H = -\frac{1}{K} + \frac{1}{F}$ but these do not work because the former is backward elliptic and the latter does not have the true solution as a stable steady state solution, by the same reasoning as for the case $H = K - F$.

5.3.2 Seeking A Related Solution

Because of the above failures, we instead change the problem and seek only a related solution to the Minkowski Problem. We make the guess

$$H = \frac{K}{F} - \lambda,$$

where $\lambda = \lambda(\phi)$ is a functional to be chosen later to prevent unstable steady state solutions. Note the solution of $H = 0$, call it ψ, is no longer a solution of the Minkowski Problem. In fact, noting that $|\nabla\psi| = 0$ happens almost nowhere and away from the zero level set of ψ, for all intents and purposes the zero level set for the steady state solution satisfies $K = CF$, where C is a constant. So on the zero level set, the Gauss-Kronecker curvature is a constant multiple of the prescribed curvature. This means a dilation of the zero level set of ψ will give the solution to the Minkowski Problem. So steady state solutions of the partial differential equation with the above choice of H are dilations of solutions of the Minkowski Problem. From now on, we will deal only with dilated solutions of the Minkowski Problem obtained from $H = 0$.

With this choice of H, the partial differential equation takes the form

$$\phi_t = \left(\frac{K(\nabla^2\phi, \nabla\phi)}{F\left(-\frac{\nabla\phi}{|\nabla\phi|}\right)} - \lambda(\phi) \right) |\nabla\phi|.$$

The first term on the right hand side, after $|\nabla\phi|$ is multiplied into the parenthesis, is degenerate elliptic and the second term is hyperbolic.

As for λ, we choose it to be the Lagrange multiplier that preserves the enclosed volume of the zero level set of ϕ. So, with H_0 denoting the Heaviside function, λ can be derived from

$$\begin{aligned} 0 &= \frac{d}{dt}\int_{\mathbf{R}^{n+1}} H_0(-\phi)dx \\ &= -\int_{\mathbf{R}^{n+1}} \phi_t\delta(\phi)dx \\ &= -\int_{\mathbf{R}^{n+1}} \left[\frac{K}{F} - \lambda\right] \delta(\phi)|\nabla\phi|dx \\ &= -\int_{\mathbf{R}^{n+1}} \frac{K}{F}\delta(\phi)|\nabla\phi|dx + \lambda\int_{\mathbf{R}^{n+1}} \delta(\phi)|\nabla\phi|dx, \end{aligned}$$

giving

$$\lambda(\phi) = \frac{\displaystyle\int_{\mathbf{R}^{n+1}} \frac{K(\nabla^2\phi, \nabla\phi)}{F(-\frac{\nabla\phi}{|\nabla\phi|})}\delta(\phi)|\nabla\phi|dx}{\displaystyle\int_{\mathbf{R}^{n+1}} \delta(\phi)|\nabla\phi|dx}.$$

With this choice of λ, the steady state solution should be a stable steady state solution since the λ prevents surfaces from shrinking to nothing or growing uncontrolled. Also this solution will be, in a sense, close to the initial shape. This is because the initial shape need only converge to the dilation of the solution of the Minkowski Problem that has equal volume. Other choices of λ, such as one that keeps surface area fixed, are also possible. Numerically, we also note the preserved volume condition keeps the final surface well resolved, which is an advantage.

As of yet, there is no theoretical justification for convergence or even preservation of strict convexity for our method. But numerical results show that we get convergence in all situations tested.

5.4 The Numerical Method

A uniform grid is placed over $\mathbf{R}^3$. This facilitates all finite difference scheme constructions and calculations. The partial differential equation we are discretizing has a degenerate elliptic term and a hyperbolic term on the right hand side. We use second order central differencing on the spacial derivatives of the degenerate elliptic term and Hamilton-Jacobi WENO on those of the hyperbolic term. The time derivative is discretized using third order TVD-RK.

The Gauss-Kronecker curvature K is computed by multiplying together the nonzero eigenvalues of the matrix

$$\frac{1}{|\nabla\phi|}P_{\nabla\phi}\nabla^2\phi P_{\nabla\phi},$$

where $P_v = I - \frac{v \otimes v}{|v|^2}$ is the orthogonal projection matrix, projecting vectors onto the plane with normal vector v, and I is the identity matrix. Note this procedure for $n = 2$ is relatively easy since the characteristic polynomial for the above matrix is of degree 3 and has zero for a root. So the nonzero eigenvalues are roots of a second degree polynomial. The quadratic formula can then be used to find these roots. In higher dimensions, other methods can be used to find the eigenvalues of the matrix.

For λ, the integral is approximated using a second order method and the delta function is replaced by a numerical delta function. Also, all derivatives inside the integrals are approximated using second order central differencing.

Not all the points in $\mathbf{R}^3$ need to be used since we are only interested in the zero level set of ϕ. We can use a local level set method[25] to perform computations only in a neighborhood around the zero level set. This neighborhood is defined as all points of a certain distance, usually a constant times the spacial step size Δx, away from the zero level set. In conjunction with this, the level set function is reinitialized to a signed distance function at every time step to prevent errors caused by the boundary of the neighborhood from propagating to the zero level set. This is done by iterating a few steps of the partial differential equation

$$\tilde{\phi}_t + sgn(\tilde{\phi}_0)(|\nabla\tilde{\phi}| - 1) = 0,$$

after each time step of the main equation. Here, $\tilde{\phi}_0$ is taken to be the ϕ generated after each time step. The steady state solution to this reinitialization equation satisfies $|\nabla\tilde{\phi}| = 1$ and has the same zero level set as $\tilde{\phi}_0$, hence giving signed distance. Enforcing signed distance also prevents steep and flat gradients in the level set function that can break down the finite difference approximations. Actually, signed distance is probably not the best form for the level set function since it may cause kinks or lose strict convexity on the nonzero level sets. However, this will only happen away from the zero level set surface so if the grid is properly refined, we can avoid this circumstance. Still, a smoother form would be better but this may lead to difficulties in the localization algorithm. For now, we just set the Gauss-Kronecker curvature to zero if the approximation takes a negative value.

Also note that steady state may only occur at the zero level set, not necessarily at the other level sets, and so our stopping condition needs to take this into account. We compare the signed distance functions near the zero level sets at two different times. If the difference is smaller than a certain tolerance, then we stop and claim convergence. Note, the use of the signed distance function means we are only comparing the zero level sets since the signed distance function is uniquely determined by its zero level set.

Another consideration is that a prescribed curvature F satisfying the compatibility condition may not be easy to construct. Generating it from a given shape may require a lot of effort. A simple way to get around this is to look at F that are symmetric with respect to each coordinate plane. Then F automatically satisfies the compatibility condition.

5.4.1 Numerical Results

We applied this method to many different prescribed curvatures $F > 0$ satisfying the compatibility condition. The initial condition ϕ_0 was always taken to be a ball. In each case, ϕ reached steady state, according to our convergence criterion, and the zero level set of this function was a dilated soln of the Minkowski Problem under the corresponding F. We considered only the case $n = 2$ so the hypersurfaces could be plotted but the method works for any n.

1. We choose F from the Gauss-Kronecker curvature for an ellipse with principal radii of 3, 2, and 1. The expression can be derived from [12]. The result is plotted in Figure 5.1 on a 50 by 50 by 50 grid.

2. We choose F to approximate symmetric cones. The parts of the cones with zero Gauss-Kronecker curvature are changed to have a small positive Gauss-Kronecker curvature. Similarly, the parts of the cones with inifinite Gauss-Kronecker curvature are changed to have large Gauss-Kronecker curavature. Thus the cones are slightly smoothed out. The result is plotted in Figure 5.2 on a 50 by 50 by 50 grid.

3. We choose F from an egg shape whose top is an ellipse and bottom is a sphere. The Gauss-Kronecker curvatures can be easily obtained using this fact. The result is plotted in Figure 5.3 on a 50 by 50 by 50 grid.

4. We choose F to be a slightly lopsided sphere. This comes from

   $$F(x) = \frac{C_1}{C_2 - x_1^2},$$

   where C_1 and C_2 are constants and x_1 is the first component of x. The result is plotted in Figure 5.4 on a 50 by 50 by 50 grid.

5. We choose F to approximate a cylinder. Once again the cylinder is slightly smoothed out to avoid zero or infinite Gauss-Kronecker curvature. The result is plotted in Figure 5.5 on a 50 by 50 by 50 grid.

6. We choose F from a clam shape coming from the union of two spheres, with the parts with infinite Gauss-Kronecker curvature slightly smoothed out. The result is plotted in Figure 5.6 on a 50 by 50 by 50 grid.

5.5 Conclusion

We constructed a partial differential equation whose steady state solution, when dilated, is a solution to the Minkowski Problem. This equation was derived using a level set approach to represent hypersurfaces. The main point is preserving the volume enclosed by the hypersurface in order to prevent it from shrinking to nothing or growing uncontrolled. This is accomplished using standard level set tools such as the Lagrange multiplier. Further tools are also used to localize the method and force the level set function to be nice. So the level set approach allows us to easily write down the form of the equation. Discretization is also easy and we use standard finite difference schemes. Finally, numerical results show that our method can generate dilated solutions to the Minkowski Problem.

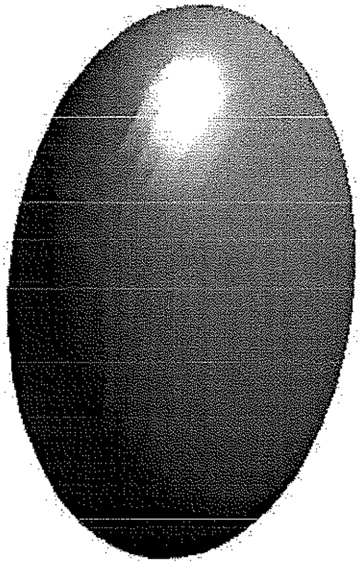

Figure 5.1: This is an ellipse with principal radii of 1 in the x-direction, 2 in the y-direction, and 3 in the z-direction.

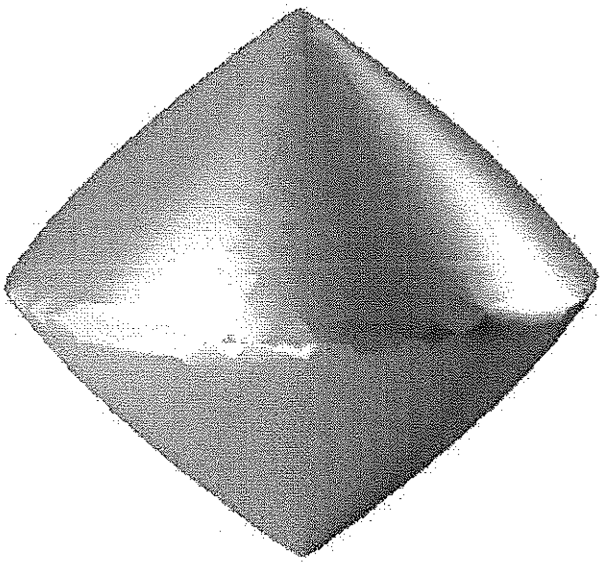

Figure 5.2: This is the union of two approximate cone shapes. They are constructed from making the Gauss-Kronecker curvature positive and small whenever it should be zero in a real cone.

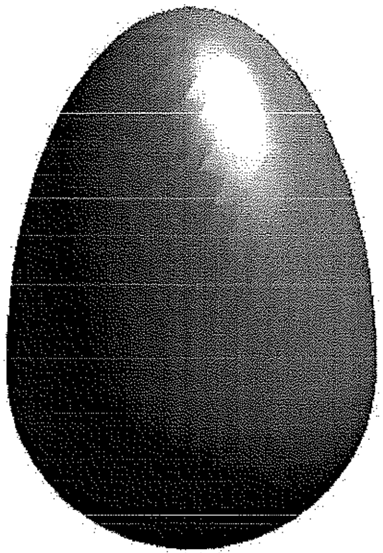

Figure 5.3: This is the union of an ellipse and a sphere, creating an egg shape. This is accomplished by prescribing the Gauss-Kronecker curvature to be that of the ellipse for inward normals that have negative z-components and that of a sphere otherwise.

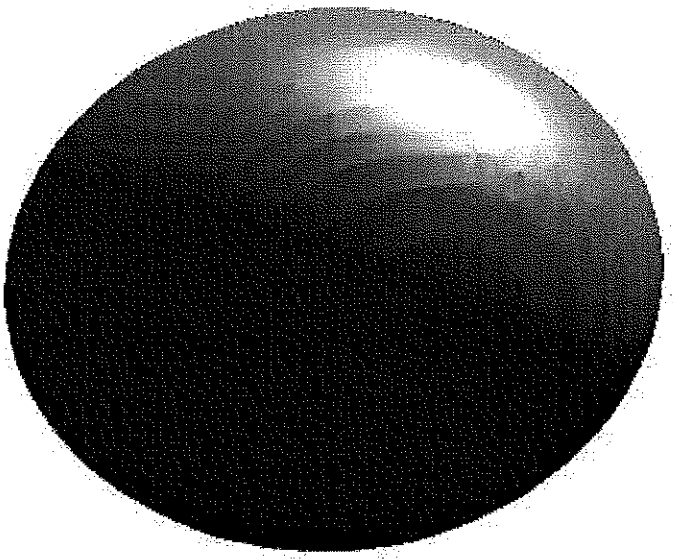

Figure 5.4: This is a slightly lopsided sphere.

Figure 5.5: This is an approximate cylinder. It is constructed by making the Gauss-Kronecker curvature positive and small whenever it should be zero in a real cylinder.

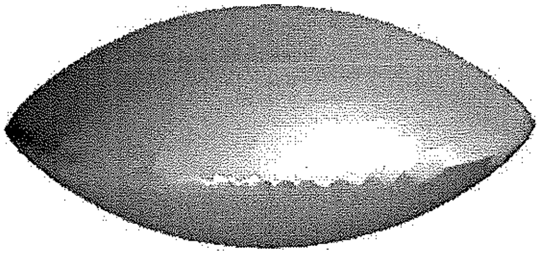

Figure 5.6: This is a clam shape derived from the union of two spheres. This is constructed by prescribing the Gauss-Kronecker curvature to be some constant if the angle made by the inward normal with the z-axis is some value and a large constant otherwise.

CHAPTER 6

A Variational Based Level Set Approach for Constructing Wulff Minimal Surfaces

6.1 Abstract

Wulff minimal surfaces arise in many physical settings. They appear in constrained crystal growth and also as minimal surfaces, where the surface is a soap film with a given wire boundary. They contribute to the studies of crystal structures and growth and, as minimal surfaces, to the studies of flexible and inextensible films. They can also be found in areas such as biology, medical technology, translation nets, relativity theory, and architecture (see [8]).

The Wulff minimal surface problem consists of finding a surface of least surface energy attached to a given boundary. This is most clearly seen in the minimal surface case where we have a soap film that is of least surface area passing through a given wire boundary. The problem stated more rigorously is: given a finite union Θ of curves in $\Re^3$ and given a convex surface tension $\gamma : S^{n-1} \to \mathbf{R}_+$, find a surface Γ minimizing the surface energy

$$E(\Gamma) = \int_\Gamma \gamma(\vec{n})ds,$$

passing through Θ. So E is the surface integral of $\gamma(\vec{n})$, where $\vec{n}$ is the unit outward normal vector. Note E may have more than one local minimum and so the minimizing surface Γ does not have to be unique. We also note if $\gamma \equiv 1$,

the energy is just the surface area of Γ and surfaces minimizing this are minimal surfaces. The Wulff minimal surface problem is easily restated and valid in $\Re^2$ and other dimensions.

6.2 Other Methods

Existing methods for solving this problem are usually very complicated. One such is by David Chopp[8]. He employs the level set method, as introduced by Osher and Sethian [24], to compute minimal surfaces with a given boundary. However, much work has to be done with the boundary before executing the program and the algorithm itself is complicated. Jean Taylor[31] has a different method that can handle Wulff shapes. However the algorithm is complicated and, since it is not a level set based method, does not nicely handle topological changes, i.e., merging. Merging phenomena are observed, for example, for soap film when the boundary wire is two circles that are far apart. The soap film will touch itself and snap off to form a piece of film in each circle. Both methods require initial shapes to be given that pass through the boundary. However, when the boundary is sufficiently complicated, finding valid initial shapes is difficult. So the main problems with current algorithms are from the following: handling complicated boundaries, topological changes, and general surface energies.

We want to derive a simpler method that can easily handle a variety of given boundaries. It will be based on a gradient descent flow minimizing a certain energy and will employ a local level set method[25]. It should be noted that this method does not find all minimal Wulff surfaces. Unstable minimal surfaces are especially difficult to construct because of their instability. Also, only convex energies are considered here.

6.3 Level Set Formulation of Problem

We first study the level set formulation of the problem. Let ϕ be a real valued function over the space $\mathbf{R}^3$ that also depends on time. This will be our level set function and the zero level set, $\Gamma = \{\vec{x} \in \Re^3 | \phi(\vec{x}, t) = 0\}$, of ϕ will be the surface of interest. Evolving ϕ instead of Γ gives several advantages. We do not have to track individual points on Γ. More importantly, any topological changes are easily and automatically handled when the surface tries to pass through itself. Note that the outward normal vector is

$$\vec{n} = \frac{\nabla \phi}{|\nabla \phi|}.$$

We first extend γ to $\mathbf{R}^3$,

$$\gamma(\vec{x}) = |\vec{x}| \gamma \left(\frac{\vec{x}}{|\vec{x}|} \right),$$

for $\vec{x} \in \mathbf{R}^3$. We may then rewrite the problem as follows: find a level set function ϕ minimizing the surface energy of the zero level set Γ,

$$E_\Gamma(\phi) = \int \gamma \left(\frac{\nabla \phi}{|\nabla \phi|} \right) \delta(\phi) |\nabla \phi| dx,$$

with Γ passing through Θ. Here δ denotes the one dimensional delta function. So the associated gradient descent equation for Γ becomes

$$\phi_t = \nabla \cdot \left(\nabla \gamma \left(\frac{\nabla \phi}{|\nabla \phi|} \right) \right) |\nabla \phi|,$$

away from Θ, and the steady state surface satisfies

$$\nabla \cdot \left(\nabla \gamma \left(\frac{\nabla \phi}{|\nabla \phi|} \right) \right) = 0,$$

away from Θ. This gives the correct behavior for the surface away from the boundary. Note in the case of minimal surfaces,

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

and so level sets move by mean curvature

$$\kappa = -\nabla \cdot \left(\frac{\nabla \phi}{|\nabla \phi|} \right).$$

The steady state zero level set surface thus has zero mean curvature.

6.4 Deriving an Energy Satisfying the Constraint

We now devise a related energy that will automatically fix Θ at the boundary of Γ. Let $d_\Theta(\vec{x})$ denote the distance from $\vec{x}$ to Θ and let μ be a smooth increasing function mapping $\mathbf{R} \rightarrow \mathbf{R}_+ \cup \{0\}$ and satisfying

$$\mu(r) = \begin{cases} 0 & \text{if } r \leq \epsilon_1 \\ 1 & \text{if } r > \epsilon_2, \end{cases}$$

where $0 < \epsilon_1 < \epsilon_2$. We then define the related energy

$$E_\mu(\phi) = \int \mu(d_\Theta) \gamma \left(\frac{\nabla \phi}{|\nabla \phi|} \right) \delta(\phi) |\nabla \phi| dx,$$

and so the associated gradient flow equation takes the form

$$\phi_t = \nabla \cdot \left(\mu(d_\Theta) \nabla \gamma \left(\frac{\nabla \phi}{|\nabla \phi|} \right) \right) |\nabla \phi|.$$

Note that

$$\phi_t(\vec{x}, t) \equiv 0,$$

in the set $\{\vec{x} \in \Re^3 | d_\Theta(\vec{x}) < \epsilon_1\}$ and so ϕ will not change near Θ. This means that if we start with a level set function whose zero level set, Γ, passes through Θ, then Θ will remain in Γ for all time. Thus the constraint is satisfied. Also

$$\phi_t = \nabla \cdot \left(\nabla \gamma \left(\frac{\nabla \phi}{|\nabla \phi|} \right) \right) |\nabla \phi|,$$

in the set $\{\vec{x} \in \Re^3 | d_\Theta(\vec{x}) > \epsilon_2\}$ and so the level sets of ϕ move with the correct velocity away from Θ. However, in between these two regions, the velocity can be strange. In fact, the steady state of the gradient descent flow satisfies

$$\nabla \cdot \left(\mu(d_\Theta) \nabla\gamma \left(\frac{\nabla\phi}{|\nabla\phi|} \right) \right) |\nabla\phi| = 0,$$

and so we get a balancing of terms

$$\nabla(\mu(d_\Theta)) \cdot \left(\nabla\gamma \left(\frac{\nabla\phi}{|\nabla\phi|} \right) \right) + \mu(d_\Theta) \nabla \cdot \left(\nabla\gamma \left(\frac{\nabla\phi}{|\nabla\phi|} \right) \right) = 0,$$

in between the regions. This in no way guarantees the correct shape for steady state solutions. Also numerically, ϵ_2 needs to be rather large and so we may not get the desired shape in a rather large region.

6.5 Modified Equation for Correct Steady State

To remedy this, we simply take $\mu(d_\Theta)$ out of the divergence in the equation. So the equation becomes

$$\phi_t = \mu(d_\Theta) \nabla \cdot \left(\nabla\gamma \left(\frac{\nabla\phi}{|\nabla\phi|} \right) \right) |\nabla\phi|.$$

We now note that

$$\phi_t(\vec{x}, t) \equiv 0,$$

in the set $\{\vec{x} \in \Re^3 | d_\Theta(\vec{x}) \leq \epsilon_1\}$, so we may in fact allow $\epsilon_1 = 0$ and still keep Γ fixed at Θ. Also we still have

$$\phi_t = \nabla \cdot \left(\nabla\gamma \left(\frac{\nabla\phi}{|\nabla\phi|} \right) \right) |\nabla\phi|,$$

in the set $\{\vec{x} \in \Re^3 | d_\Theta(\vec{x}) > \epsilon_2\}$. So the proper motion of the surface is preserved in the outer region. In between the two regions, steady state solutions now satisfy

$$\mu(d_\Theta) \nabla \cdot \left(\nabla\gamma \left(\frac{\nabla\phi}{|\nabla\phi|} \right) \right) = 0,$$

and since $\mu(d_\Theta)$ is never zero here, this means

$$\nabla \cdot \left(\nabla\gamma \left(\frac{\nabla\phi}{|\nabla\phi|} \right) \right) = 0.$$

So steady state solutions will satisfy the correct property in between regions. Therefore, we see that the steady state solutions will satisfy the correct property in the set $\{\vec{x} \in \Re^3 | d_\Theta(\vec{x}) > 0\} = \Re^3 \setminus \Theta$, where we have taken $\epsilon_1 = 0$, and if we let Γ pass through Θ initially, then the boundary constraint will also be satisfied. So the method becomes: solve to steady state

$$\begin{cases} \phi_t = \mu(d_\Theta)\nabla \cdot \left(\nabla\gamma \left(\frac{\nabla\phi}{|\nabla\phi|} \right) \right) |\nabla\phi| \\ \phi(\vec{x}, 0) = \phi_0(\vec{x}), \end{cases}$$

where $\Theta \subseteq \{\vec{x} \in \Re^3 | \phi_0(\vec{x}) = 0\}$. Note that with $\mu(d_\Theta)$ outside of the divergence, the equation resembles a projection method on the velocity of the surface. We also note that steady state is usually only achieved at the zero level set and not for the other level sets. However, we are only interested in the zero level set and it solves, in steady state, the Wulff minimal surface problem.

6.6 Numerical Considerations

To create a numerical method, we first lay down a uniform spatial grid. We take as input just the discrete points approximating the boundary curves Θ, the formula for the gradient of the surface tension function, $\nabla\gamma$, and the initial shape ϕ_0. We can also make do with just γ given instead of $\nabla\gamma$ since $\nabla\gamma$ can be numerically generated from γ. We compute the distance function, $d_\Theta(\vec{x})$, by finding the minimum of the distances from each point on the boundary curves to the grid point $\vec{x}$, for each grid point. These values need only be computed once in the beginning and are then stored for later use.

For the initial shape, we can pick ϕ_0 so that the zero level set of ϕ_0 passes through Θ. In most cases though, we do not even have to find such an initial level set function. We can simply start with a ϕ_0 whose zero level set encloses Θ in its interior. An example of such a level set function is $\phi_0(\vec{x}) = |\vec{x}|^2 - r^2$. The zero level set of this function is a sphere with radius r and for r large enough, the boundary curves will be enclosed within the sphere. From this, what will happen is the sphere will shrink and when it hits a point in Θ, it will be held back at that point. This greatly simplifies the search for an adequate initial level set function.

Another way to get an initial shape that passes through Θ is by using an interpolation program such as the one by [35]. We note that their q-form energy is equivalent to minimizing the surface area while trying to satisfy the constraint of the surface passing through certain given points. This is formulated using a penalty method approach.

Because there are errors in the representation of Θ and because of the discrete nature of the method, we must insure that ϕ will not break away from or pass through Θ by taking $\epsilon_1 > 0$. In practice, we usually take ϵ_1 to be around Δx so that there is at least one grid point inside the ball of radius ϵ_1 around each boundary point. This is equivalent to thickening the boundary curves into tubes of radius ϵ_1. Since ϕ will be held still at all grid points in these tubes, Γ cannot escape the boundary. The choice for ϵ_2 is not as important since ϵ_2 does not affect the steady state solutions but only changes the convergence rate in the neighborhood between ϵ_1 and ϵ_2. This defines μ.

We are now free to use an explicit method to solve the gradient descent equation. We use second order central differencing on all spacial derivatives and third order TVD-RK in time. This leads to a CFL restriction that Δt needs to be less than a constant times Δx^2.

Also, computations are only needed near Γ since this is the surface of interest. This is done by using a local level set method, as in [25]. In this method, points that are close to Γ, as measured by distance, are marked. Computations are only done at these points. The method is efficient because the distance we choose for marking the points is proportional to Δx. This process also requires ϕ to be a signed distance function. This is to prevent errors caused by the boundary of the marked points from seeping into the calculations at Γ. It also prevents the level sets of ϕ from bunching up near Θ, called the tentpole phenomenon. This bunching up, which was noticed by Chopp[8] in his paper, leads to inaccurate computations since the magnitude of the gradient of ϕ becomes large. The signed distance requirement, however, forces the level sets of ϕ to be well spaced, satisfying $|\nabla\phi| = 1$ everywhere. This can be enforced by iterating to steady state the equation derived by Sussman, Smerka, and Osher[30],

$$\tilde{\phi}_t + sgn(\tilde{\phi}_0)(|\nabla\tilde{\phi}| - 1) = 0,$$

at each time step, where $\tilde{\phi}_0$ is the ϕ derived at each time step. We, however, will not solve this to steady state but will just do a few iterations of it every time step. This will keep the level sets of ϕ relatively well spaced, especially near Γ.

In Figure 6.1, the boundary curves are two squares and the minimal surface has a catenoid shape. The whole zero level set is shown. For this problem, we fixed the parts of the zero level set surface that extend above and below the top and bottom squares, respectively. In post processing, the plotter can erase these parts of the surface to get a better picture. In Figure 6.2, the boundary is two parallel but angled circles and the minimal surface is shown. The whole zero level set surface is plotted here. The frizzled look at the boundary curves comes from the triangulating plotter applied to the moderately sharp edges there. In Figure 6.3, we look for a minimal surface where the boundary is three squares, a large one on top and two small ones that meet at a corner on the bottom. The whole zero level set was used and is shown. In Figure 6.4, we look for the minimal surface of a boundary consisting of three circles. The whole zero level set is plotted but some of it, the parts above the circles, was not used, meaning it was held fixed, in the algorithm. Thus we rounded surfaces above the circles when we should get flat surfaces. In Figure 6.5, we look for the minimal surface of a boundary that is a rectangle bent at two of its ends. The whole zero level set is shown. Note the parts of the surface extending out of the boundary are extraneous and were fixed in the algorithm.

For Wulff minimal surfaces, we used a smoothed out version of $\gamma(\vec{x}) = |x_1| + |x_2| + |x_3|$. The exact expression is

$$\gamma(\vec{x}) = \sqrt{x_1^2 + \epsilon^2} + \sqrt{x_2^2 + \epsilon^2} + \sqrt{x_3^2 + \epsilon^2},$$

for $\epsilon = 0.1$. The Wulff shape arising from this γ is a smoothed out cube, i.e., having slightly rounded faces and edges. Figure 6.6 shows the Wulff minimal surface passing through three circles, as in Figure 6.4. The whole zero level set is shown here. Note the squarish aspect of the surface, especially the indentations. Finally, in Figure 6.7, we show the Wulff minimal surface corresponding to a boundary of two parallel circles. In the minimal surface case, the resulting surface would be a catenoid. In this case, we have a surface that is more like a square.

All the algorithms for the pictures were run on 80 by 80 by 80 grids except for the minimal surface for three squares, Figure 6.3, which was run on a 100 by 100 by 100 grid.

6.7 Other Uses

We can apply the basic concepts for finding Wulff minimal surfaces to other problems such as finding the convex hull of a set of points, finding the Wulff minimal surface through a boundary subject to a fixed volume constraint, and splitting a domain into two pieces of equal volume by a Wulff minimal surface that passes through a prescribed curve on the boundary of the domain.

6.7.1 Convex Hull

Using the basic principles of the above method, we look for the convex hull of a set of points. We denote the set of points using Θ. We also note that the boundary of the convex hull's minimal principal curvature is zero at each point of the surface. Thus we construct the partial differential equation

$$\phi_t = \mu(d_\Theta)\lambda_1|\nabla\phi|,$$

where λ_1 is the minimal principal curvature and μ and d_Θ are the same as above. So if we start with an initial level set function whose zero level set encloses Θ in its interior, then the steady state Γ should satisfy $\lambda_1 = 0$ with Γ wrapped around $\mu = 0$. This is actually the convex hull for the set of points satisfying $\mu = 0$, giving an approximate convex hull for Θ. As before, the μ term keeps the surface from passing through Θ. We remark that the $|\nabla\phi|$ on the right hand side of our partial differential equation will make the level sets of ϕ move in the inward normal direction with speed $\mu(d_\Theta)\lambda_1$.

We calculate λ_1 by solving for it as the smallest nonzero eigenvalue of the matrix

$$\frac{1}{|\nabla\phi|}P_{\nabla\phi}\nabla^2\phi P_{\nabla\phi},$$

where P_v is the orthogonal projection matrix projecting vectors onto the plane with normal v, and $\nabla^2\phi$ is the Hessian matrix of ϕ. This is a relatively simple procedure in $\mathbf{R}^3$ since we know that the above matrix has a zero eigenvalue. So we only need to solve for the smallest root of a quadratic polynomial coming from the characteristic polynomial. This can be done using the quadratic formula. In higher dimensions, other methods can be used.

Numerically, we use second order central differencing on all spacial derivatives and third order TVD-RK in time. The CFL restriction says that Δt needs to be less than a constant times Δx^2.

We show our results in Figures 6.8, 6.10, and 6.11. For Figure 6.8, we have an initial set consisting of six points, four forming a square and the other two above and below the square. Thus the convex hull should be like two pyramids stuck together. In Figure 6.9, we show the convex hull of the set of points in two spheres of different radii. In Figure 6.10, we look at the convex hull of three spheres of different radii. The resulting boundary is a surface wrapped around these three balls. In Figure 6.11, we compute the convex hull of the set of points in two linked rings. The resulting boundary is a surface wrapped around the two linked rings.

Other methods currently available, for example [28], do not handle sets with disconnected components.

6.7.2 Wulff Minimal Surfaces with Constrained Volume

In this problem, we study Wulff minimal surfaces passing through a prescribed boundary Θ with the added constraint of having a prescribed enclosed volume. Let the initial ϕ pass through the boundary and have the wanted enclosed volume. Then the problem becomes solving for the minimal Wulff surface while retaining the same volume for all time. This can be solved using a projection gradient method (see []). The partial differential equation takes the form

$$\phi_t = \mu(d_\Theta)\left(\nabla\cdot\left(\nabla\gamma\left(\frac{\nabla\phi}{|\nabla\phi|}\right)\right) - \lambda\right)|\nabla\phi|,$$

where λ is a Lagrange multiplier enforcing

$$\frac{d}{dt}\int_{\mathbf{R}^3} H(-\phi)dx = 0.$$

H here denotes the one dimensional Heaviside function. Thus λ can be solved by differentiating the Heaviside function and using the expression for ϕ_t. In fact, λ takes the form

$$\lambda = \frac{\int_{\mathbf{R}^3}\mu(d_\Theta)\nabla\cdot\left(\nabla\gamma\left(\frac{\nabla\phi}{|\nabla\phi|}\right)\right)|\nabla\phi|\delta(\phi)\delta(\psi)|\nabla\psi|dx}{\int_{\mathbf{R}^3}\mu(d_\Theta)|\nabla\phi|\delta(\phi)\delta(\psi)|\nabla\psi|dx}.$$

So the μ term holds our surface in place, the λ term enforces the volume condition, and the rest flows the surface to minimize the Wulff energy.

As before, for most cases, we can chose our initial ϕ to just have its zero level set surround the boundary points. Then we have to think of the boundary points as tiny balls with radius related to Δx.

We have programmed this up in $\mathbf{R}^2$. The same finite difference schemes are used as in the Wulff minimal surface problem for the degenerate elliptic term on the right hand side of the partial differential equation. The integrals in λ are approximated using a second order scheme and the integrands are approximated using a discrete delta function and second order central difference schemes on all spacial derivatives. The term $-\lambda|\nabla\phi|$ is then solved using WENO-Godunov of fifth order. The time derivative is once again discretized using third order TVD-RK.

We show our results in two dimensions in Figures 6.12 and 6.13. Figure 6.12 shows the initial shape that passes through the boundary points and the final shape minimizing length and preserving the enclosed volume. Note the final shape is like that of a lens. Figure 6.13 shows the initial shape surrounding the boundary points and the final shape minimizing a Wulff energy and preserving the enclosed volume. Note the final shape is like a squarish lens.

6.7.3 Domain Splitting by Wulff Minimal Surfaces

In this problem, we want to cut a given domain into two pieces of equal volume using a minimal Wulff surface passing through a prescribed curve on the domain boundary. Let d be the signed distance function for the domain, with d negative outside the domain. Let ϕ initially pass through the prescribed curve on the domain. If we require ϕ to initially cut the domain into two pieces of equal volume, then by the same reasoning as before, the equation is

$$\phi_t = H(d)\left(\nabla \cdot \left(\nabla\gamma\left(\frac{\nabla\phi}{|\nabla\phi|}\right)\right) - \lambda\right)|\nabla\phi|,$$

where λ enforces that the enclosed volume in the domain is fixed and H is the one dimensional Heaviside function. Thus we can solve for λ from this condition, by taking a time derivative of the expression for the volume, and the equation for ϕ_t. In fact, the form of λ will be the same as in the constrained volume computations except $\mu(d_\Theta)$ is replaced by $H(d)$.

We can also create another equation for this problem that can also be used to initialize ϕ for the above equation. This equation is a penalty method based equation and takes the form

$$\phi_t = H(d)\left(\nabla \cdot \left(\nabla\gamma\left(\frac{\nabla\phi}{|\nabla\phi|}\right)\right)|\nabla\phi| - \lambda\int_\Omega\left(H(-\phi) - \frac{1}{2}\right)dx\right),$$

where λ is a penalty term. So as λ gets large, the solution tries to satsify

$$\int_{\mathbf{R}^3} H(d)\left(H(-\phi) - \frac{1}{2}\right)dx = 0,$$

which is what we want. The initial ϕ is chosen to pass through the prescribed curve on the boundary of the domain. The equation then flows the curve by Wulff flow to satisfy the volume condition in steady state. For λ very large, i.e., dropping the Wulff flow term, we get a fast method for initializing the ϕ to split the domain into two regions of equal area.

We use just the initialization equation with a large λ for our simulations in two dimensions. The integral is approximated using a second order scheme and the rest is discretized the same as before. Also, H is replaced by a smoothed out numerical Heaviside function. The results are shown in Figures 6.14 and 6.15. Note the curve that splits the domain should is allowed to wander out of the domain a little but as the grid size gets smaller, the curve being in the domain is more strongly enforced. For cleaner plots, we can erase the parts of the curve that are on the outside of the domain.

6.8 Conclusion

We note that our method for Wulff minimal surfaces requires a small amount of information to be given. The rest is automatically handled by the algorithm. All types of boundaries can be considered since we only need to find $d_{\Theta}(\vec{x})$, which is easily computed and just once, for any type of boundary. Also, in many cases, the initial shape does not have to pass through the boundary. This greatly simplifies the input information. The method can be successfully used for constructing minimal surfaces and is easily extended to higher dimensions or down to two dimensions. Computer simulations have yielded many Wulff minimal surfaces, especially minimal surfaces, for many different boundary curves. The applications of the method also can handle convex hulls and a variety of other energy minimizations.

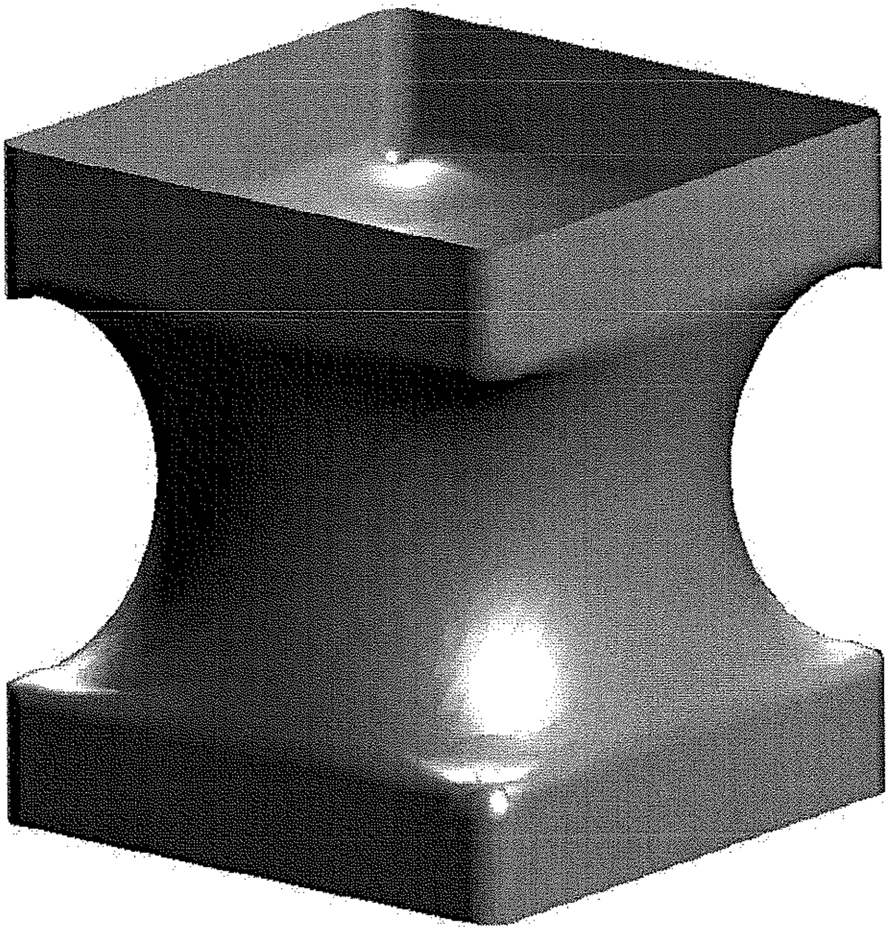

Figure 6.1: This is a minimal surface with a boundary consisting of two squares. The surface looks like a catenoid. We can cut off the parts of the zero level set surface that extend beyond the squares in post processing.

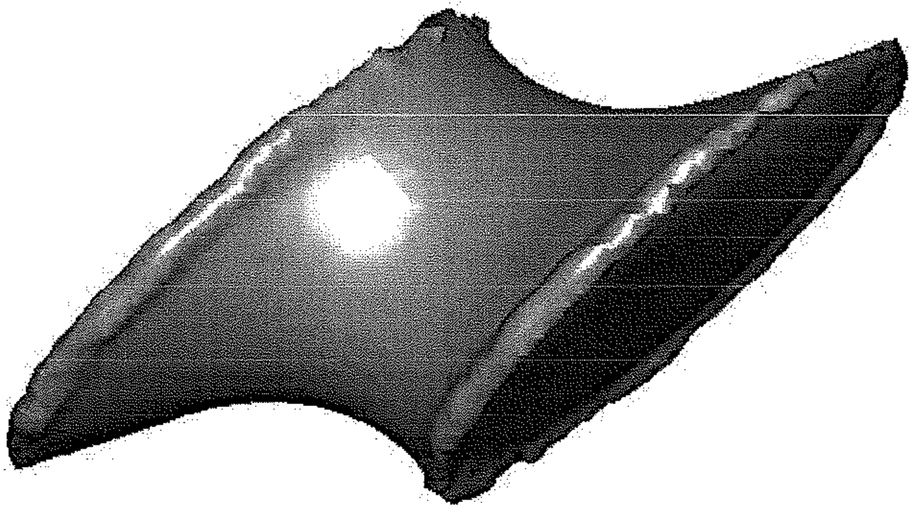

Figure 6.2: This is a minimal surface with a boundary consisting of two angled circles. The complete zero level set surface is shown here. The surface looks like a catenoid.

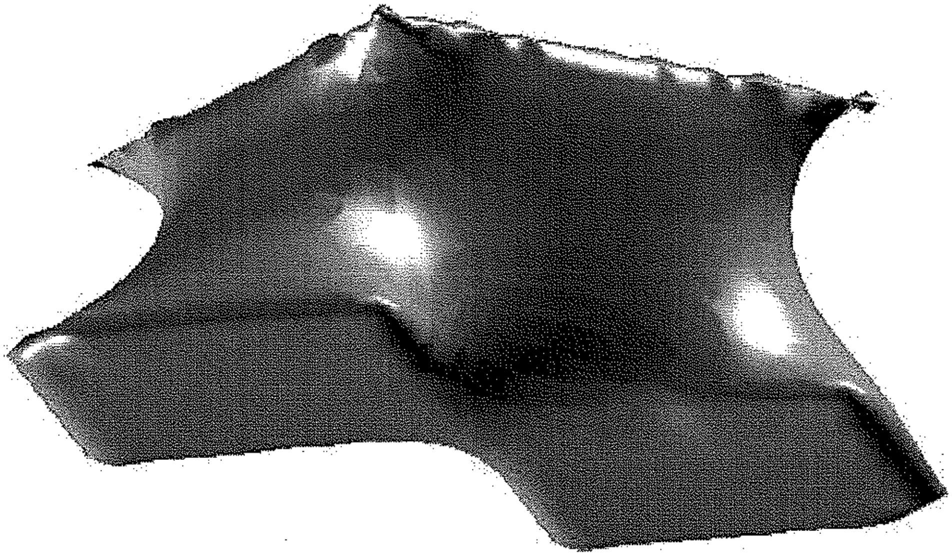

Figure 6.3: This is a minimal surface with a boundary consisting of three squares, a large one on top and two small ones that touch at a corner on the bottom. The complete zero level set surface is shown here.

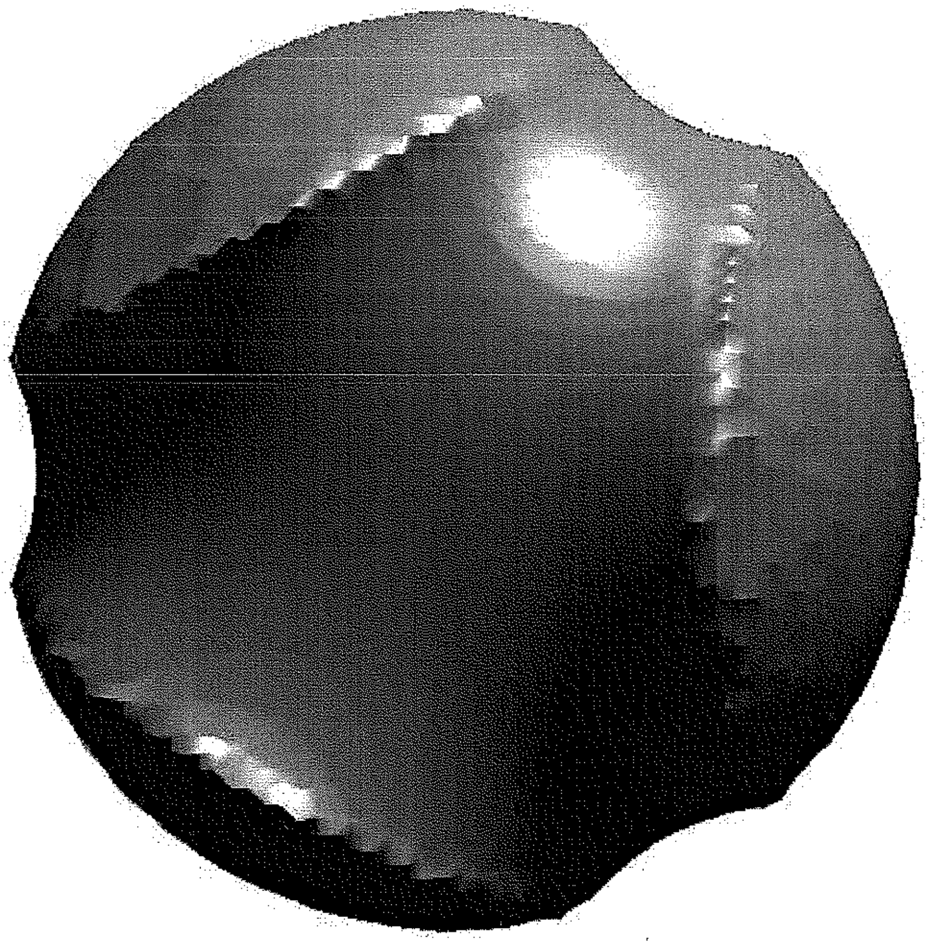

Figure 6.4: This is a minimal surface with a boundary consisting of three circles, ignoring the rounded part above each circle. The complete zero level set surface is shown here. The rounded parts that should be flat come from setting the velocity to be zero there since we were not interested in that part of the surface. It is possible to cut these parts out with the plotter.

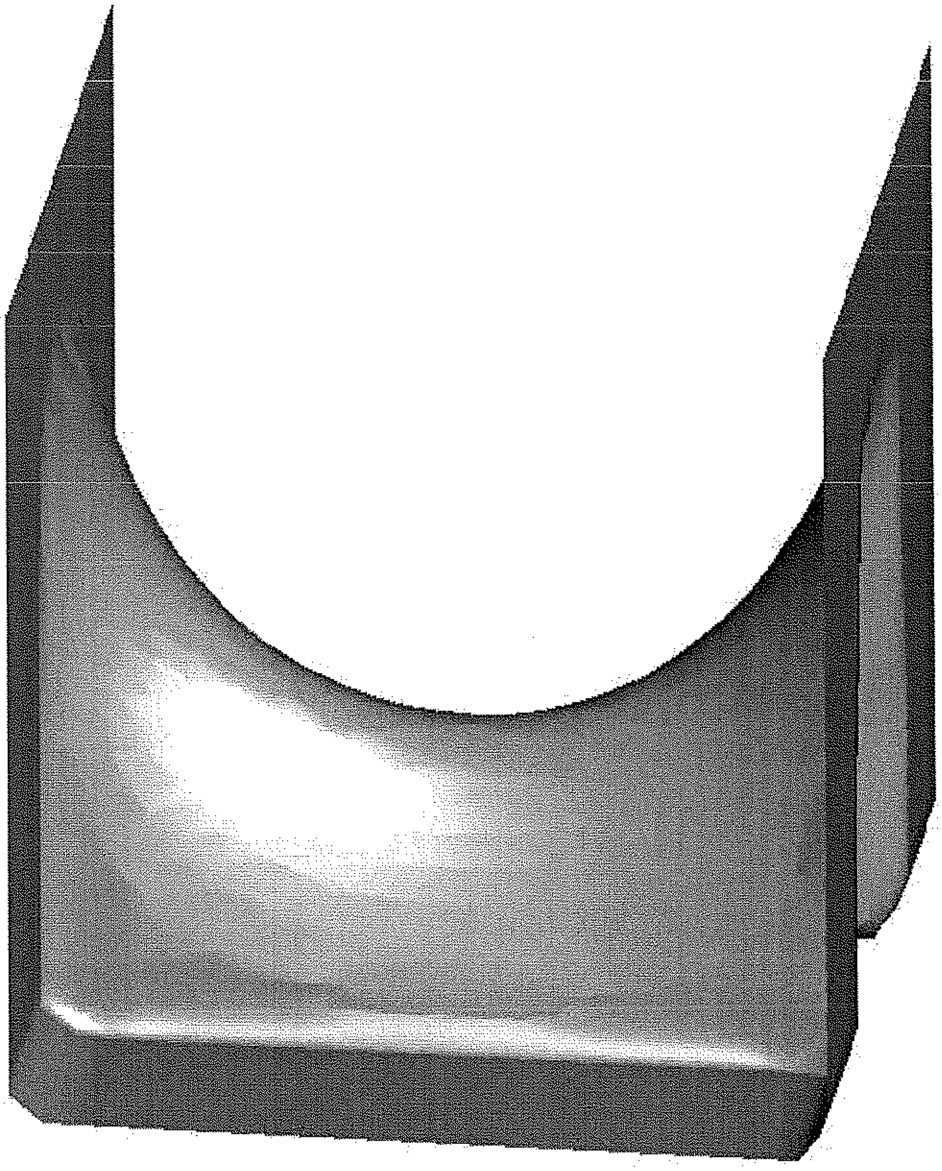

Figure 6.5: This is a minimal surface with a boundary consisting of a rectangular wire bent at two of its ends, ignoring the surface extending out of the wire. The complete zero level set surface is shown here. The extra surface parts are there for the surface to have the right topology and are not interesting. It is possible to cut these parts out with the plotter.

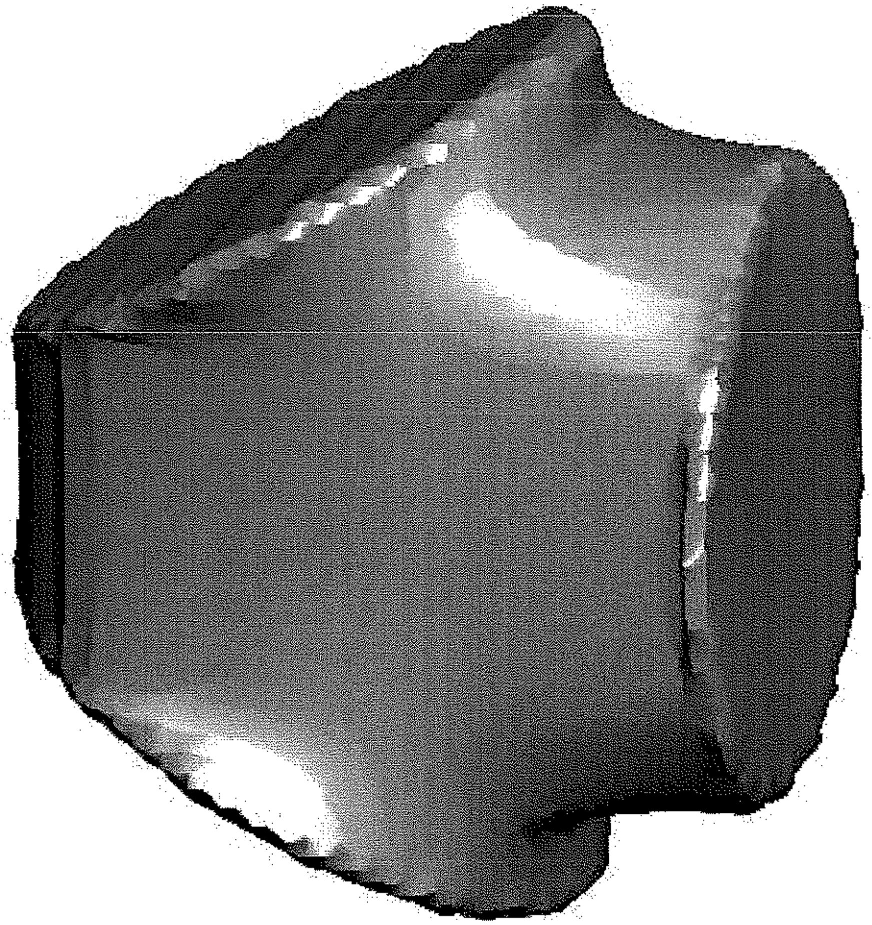

Figure 6.6: This is a Wulff minimal surface with a boundary consisting of three circles, as in Figure 6.4. The Wulff energy used is a smoothed out version of $\gamma(\vec{x}) = |x_1| + |x_2| + |x_3|$. The complete zero level set surface is shown here. Note the squarish nature of the surface.

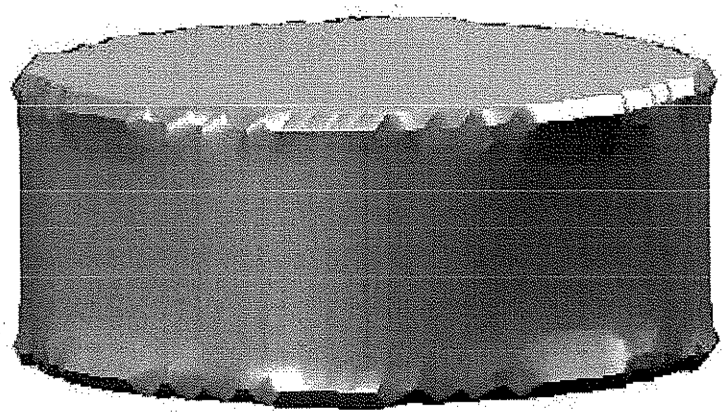

Figure 6.7: This is a Wulff minimal surface with a boundary consisting of two parallel circles. The Wulff energy used is a smoothed out version of $\gamma(\vec{x}) = |x_1| + |x_2| + |x_3|$. The complete zero level set surface is shown here. Note the squarish shape of the resulting surface. For minimal surfaces, the shape would be a catenoid.

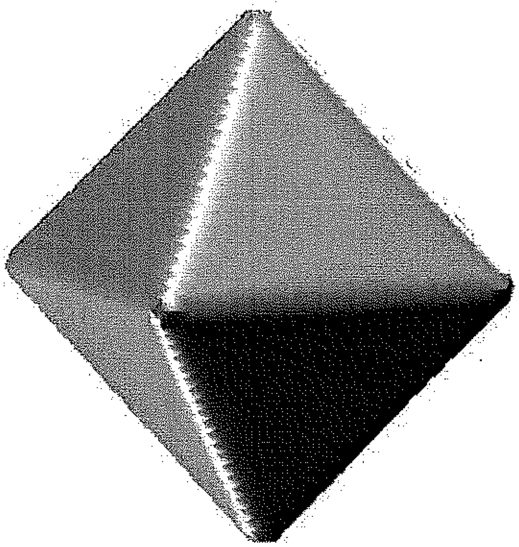

Figure 6.8: This is the boundary of the convex hull of six points. The shape is two pyramids.

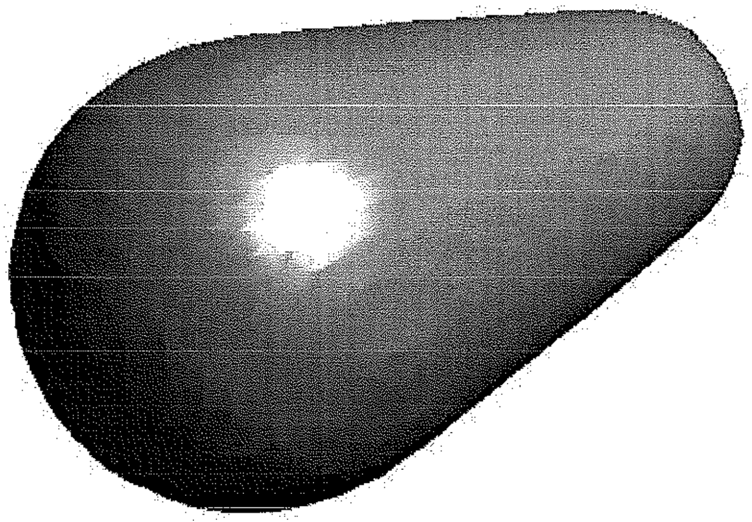

Figure 6.9: This is the boundary of the convex hull of the points in two spheres.

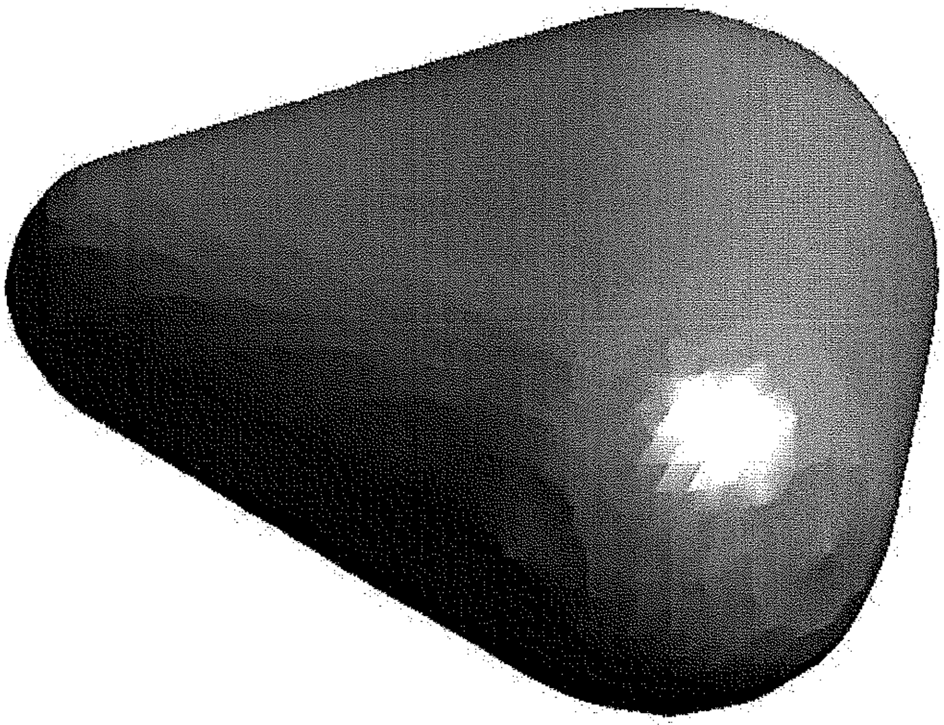

Figure 6.10: This is the boundary of the convex hull of the points in three spheres.

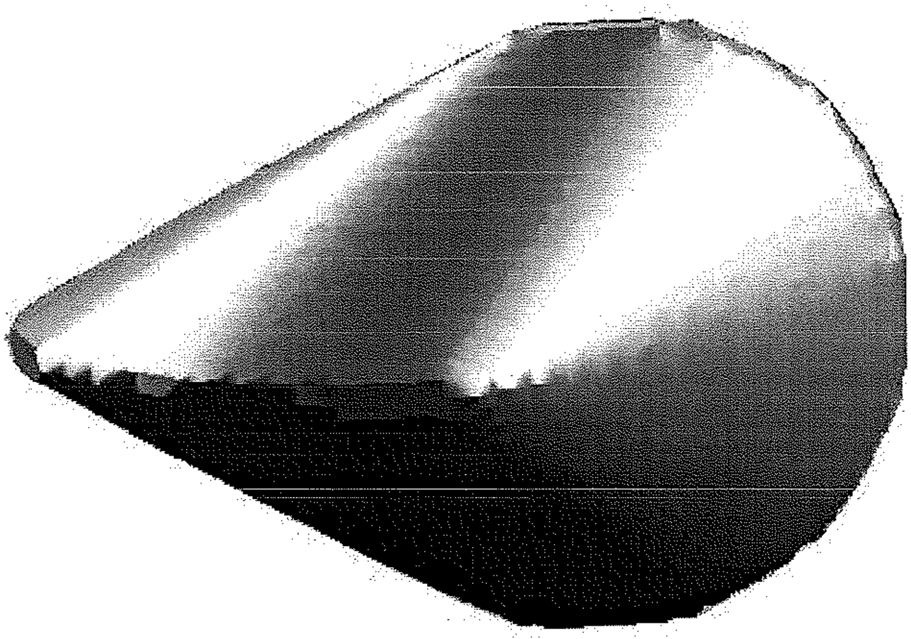

Figure 6.11: This is the boundary of the convex hull of the points in two linked rings.

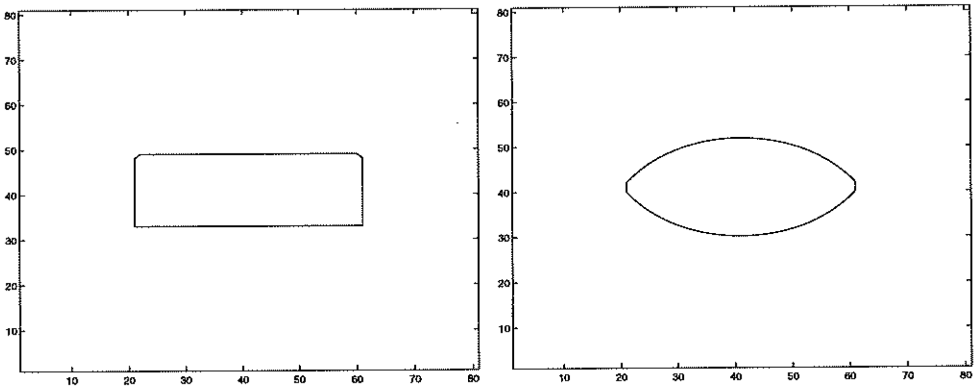

Figure 6.12: This is curvature flow with fixed enclosed volume in two dimensions of a curve passing through two boundary points. The original shape (left) is a box surrounding the boundary points. The final shape (right) is a lens that is the union of two circles.

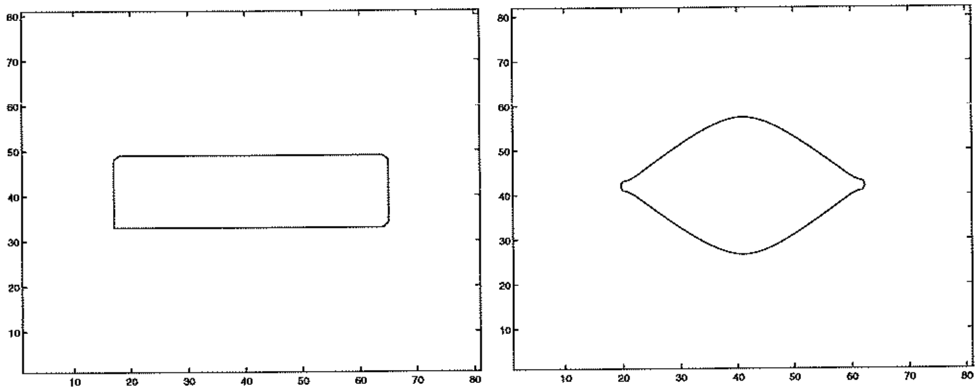

Figure 6.13: This is Wulff flow with fixed enclosed volume in two dimensions of a curve passing through two boundary points. The original shape (left) is a box passing throught the boundary points. The final shape (right) is a lens with a more squarish structure due to the Wulff energy.

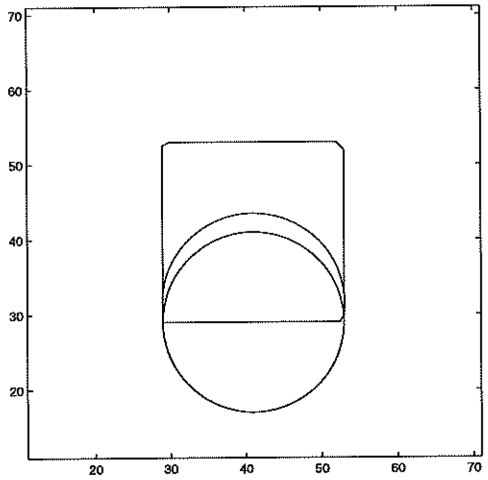

Figure 6.14: This is the splitting of a domain with the minimal length curve into two equal volume pieces. The domain here is a square and the initial curve is a circle not satisfying the volume constraint. Thus the circle expands outward into the final shape.

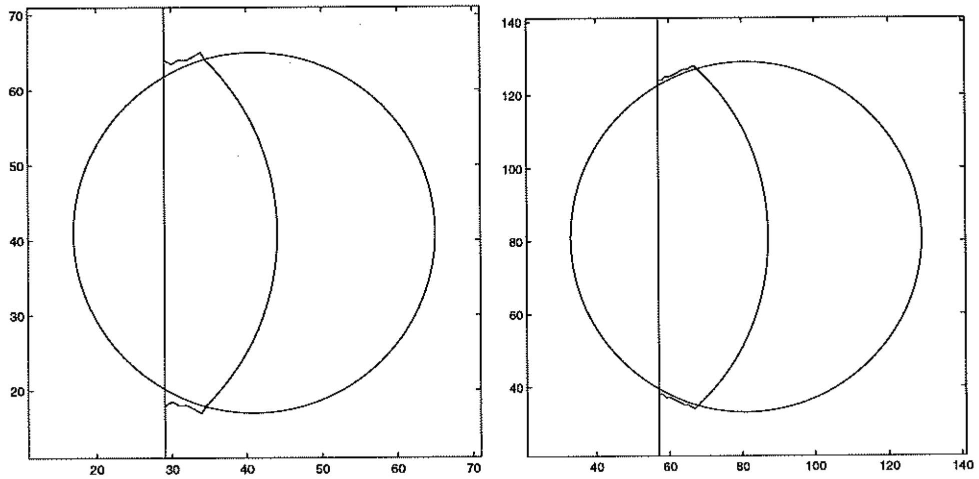

Figure 6.15: This is the splitting of a domain with the minimal length curve into two equal volume pieces. The domain here is a circle and the initial curve is a line not satisfying the volume constraint. Thus the line grows and into the final shape. On the left is the program on an 80 by 80 grid and on the right is the program on a 160 by 160 grid.

CHAPTER 7

The Level Set Method Applied to Unstable Cases in Irreversible Aggregation

7.1 Abstract

We consider cases in irreversible aggregation that produce unstable solutions and compare solutions derived using the level set method with theoretical results. We will look for dendritic formation in the computed solutions as well as compare the island boundaries, growth rates, and initial velocities with exact solutions. Results will show the level set method can reproduce the characteristics of the instability in the problem.

7.2 Setup

The level set method[24] has recently been applied to island dynamics in molecular beam epitaxy[4]. Each of the islands has a discrete height and we are interested in the shapes of their boundaries at future times. Different levels of a level set function are set to denote the island boundaries. The time evolution of the level set function then gives the location of the island boundaries at future times. We consider here only the case of a step train under the process of irreversible aggregation without nucleation and we will compare the calculations from the level set method with theoretical results.

7.3 Theoretical Results

Let the graph of $Y(y,t)$ denote the step boundary and let $\rho(x,y,t)$ be the density of adatoms on the substrate. Then the evolution of the step is given by the equations,

$$\begin{aligned}
&(\partial_t - D\nabla^2)\rho = F, && Y - L < y < Y + L \\
&\rho = 0, && y = Y - L, Y + L \\
&D\{n \cdot \nabla\rho(Y+L) - n \cdot \nabla\rho(Y-L)\} = -a^{-2}v \cdot n, && \\
&Y_t = v \cdot n, &&
\end{aligned}$$

where

$$n = \frac{(-Y_x, 1)}{\sqrt{1+Y_x^2}}$$

denotes the normal to the step and F is the flux of adatoms, $2L$ the period of the step in the y-direction, a the diameter of an adatom, and D the surface diffusion constant.

Now performing a shift, $Y = v_0 t + Y'$, to center the step, the evolution equations become, also replacing ρ by $\rho(x, y', t)$,

$$\begin{aligned}
&(\partial_t - v_0\partial_{y'} - D\nabla^2)\rho = F, && Y' - L < y' < Y' + L \\
&\rho = 0, && y' = Y' - L, Y' + L \\
&D\{n \cdot \nabla\rho(Y'+L) - n \cdot \nabla\rho(Y'-L)\} = -a^{-2}v \cdot n, && \\
&Y'_t = v \cdot n - v_0, &&
\end{aligned}$$

and we may begin a perturbation analysis of the Y' and ρ equations.

Let

$$\begin{aligned}
\rho(x, y', t) &= \rho_0(y') + \epsilon\rho_1(x, y', t) + ... \\
Y'(y', t) &= \epsilon Y'_1(y', t) + ...
\end{aligned}$$

Then the first order approximation is,

$$\rho_0 = b_0 + b_1 y' + b_2 e^{-\lambda y'}$$

where

$$\begin{aligned}
v_0 &= 2a^2 FL, \\
\lambda &= 2a^2 D^{-1} FL, \\
b_0 &= (2a^2)^{-1} \coth(\lambda L), \\
b_1 &= -(2a^2 L)^{-1}, \\
b_2 &= -(2a^2 \sinh(\lambda L))^{-1},
\end{aligned}$$

and so the second order approximation is,

$$\begin{aligned}
\rho_1 &= e^{ikx+\omega t}(\hat{\rho}_+ e^{\alpha_+ y} + \hat{\rho}_- e^{\alpha_- y}), \\
Y_1' &= \hat{Y}_1 e^{ikx+\omega t},
\end{aligned}$$

where ω, $\hat{\rho}_+$, $\hat{\rho}_-$, α_+, and α_- are derived from the equations,

$$\begin{aligned}
&(\omega + Dk^2) - v_0\alpha_+ - D\alpha_+^2 = 0, \\
&(\omega + Dk^2) - v_0\alpha_- - D\alpha_-^2 = 0, \\
&\hat{\rho}_+ e^{\alpha_+ L} + \hat{\rho}_- e^{\alpha_- L} + \hat{Y}_1 \rho_0'(L) = 0, \\
&\hat{\rho}_+ e^{-\alpha_+ L} + \hat{\rho}_- e^{-\alpha_- L} + \hat{Y}_1 \rho_0'(-L) = 0, \\
&D((\alpha_+\hat{\rho}_+ e^{\alpha_+ L} + \alpha_-\hat{\rho}_+ e^{\alpha_- L}) - (\alpha_+\hat{\rho}_+ e^{-\alpha_+ L} + \alpha_-\hat{\rho}_+ e^{-\alpha_- L})) + \\
&\quad D\hat{Y}_1(\rho_0''(L) - \rho_0''(-L)) = -a^{-2}\omega\hat{Y}_1.
\end{aligned}$$

Now since we would like $\hat{\rho}_+$, $\hat{\rho}_-$, and $\hat{Y}_1$ not all zero, this means the coefficient matrix needs to be singular and so its determinant is zero, i.e.,

$$\begin{vmatrix}
e^{\alpha_+ L} & e^{\alpha_- L} & \rho_0'(L) \\
e^{-\alpha_+ L} & e^{-\alpha_- L} & \rho_0'(-L) \\
2D\alpha_+ \sinh(\alpha_+ L) & 2D\alpha_- \sinh(\alpha_- L) & D(\rho_0''(L) - \rho_0''(-L) + a^{-2}\omega
\end{vmatrix} = 0.$$

This reduces to

$$\begin{aligned}0 = \; & (\coth(\alpha_+ L) - \coth(\alpha_- L))(D(\rho_0''(L) - \rho_0''(-L)) + a^{-2}\omega) + \\ & (\rho_0'(L) + \rho_0'(-L))D(\alpha_- - \alpha_+) + (\rho_0'(L) - \rho_0'(-L))D(\alpha_+ \coth(\alpha_- L) - \\ & \alpha_- \coth(\alpha_+ L)).\end{aligned}$$

From this equation, we see that in leading order,

$$\omega = a^2 D(\rho_0'(L) + \rho_0'(-L))|k|,$$

since $\alpha_+ = |k|$ and $\alpha_- = -|k|$ in leading order, from the first two and the last equation of the five above equations, and also $\coth(\alpha_+ L) \simeq 1$ and $\coth(\alpha_- L) \simeq -1$. Now

$$\begin{aligned}\rho_0'(L) + \rho_0'(-L) &= -\frac{1}{a^2 L \sinh(\lambda L)}(\sinh(\lambda L) - \lambda L \cosh(\lambda L)) \\ &= -\frac{e^{\lambda L}}{2a^2 L \sinh(\lambda L)}(1 + \lambda L)\left\{\frac{1 - \lambda L}{1 + \lambda L} - e^{-2\lambda L}\right\},\end{aligned}$$

but

$$\frac{1 - \lambda L}{1 + \lambda L} - e^{-2\lambda L} < 0$$

so this means that $\rho_0'(L) + \rho_0'(-L) > 0$ and so $\omega > 0$. So we have found a class of unstable solutions.

7.4 Comparing with Level Set Method Calculations

The level set method essentially considers a function ϕ, called a level set function, defined on a uniform grid in space. ϕ incorporates all island boundaries at a given height into one of its level sets, preferably one related to that height. The level set function is then updated at each time step by solving for ρ, using it to form the velocity at the island boundaries, extending this off the island boundaries to each point in the spacial grid, and then moving all level sets of ϕ by this velocity.

The end result is that at any time step, we can recover the island boundaries at a given height by finding its associated level set contours in ϕ. One advantage to this method is that it automatically treats the merging of island boundaries.

We would like the level set method to mimic the theoretical behavior in the situations we looked at above. We consider the case of initial steps $Y(y, t = 0) = \epsilon \cos(ky)$ with initial density $\rho(x, y, t = 0) = \rho_0(y) + \epsilon\rho_1(x, y, t = 0)$. These two are compatible when ϵ is small.

In the simple case of a straight step, i.e., where $\epsilon = 0$, we see that the solution derived from the level set method agrees with the theoretical solution and the step travels at the correct velocity v_0 (see Figure 7.1).

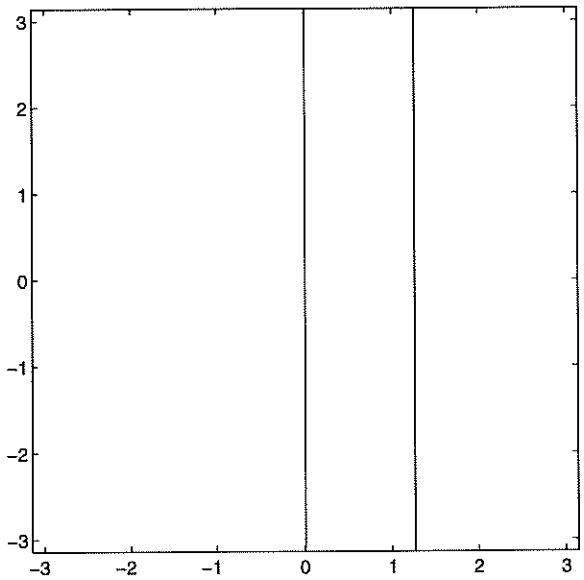

Figure 7.1: straight step, $D = 100$, $t = 0.0, 0.2$ from left to right, exact and computed solutions overlay.

But the required size of ϵ turns out to be restrictive. We see this when we compare the velocities of the step derived from Y and from ρ. In order to make the initial velocities match, ϵ must be very small (see Figure 7.3) and so the step may be under-resolved. If ϵ is chosen to be too large, then the velocities will not match. Then since the exact solution gets the step velocity from the Y equation and the level set method derives it from the ρ equation, the two solutions will not agree.

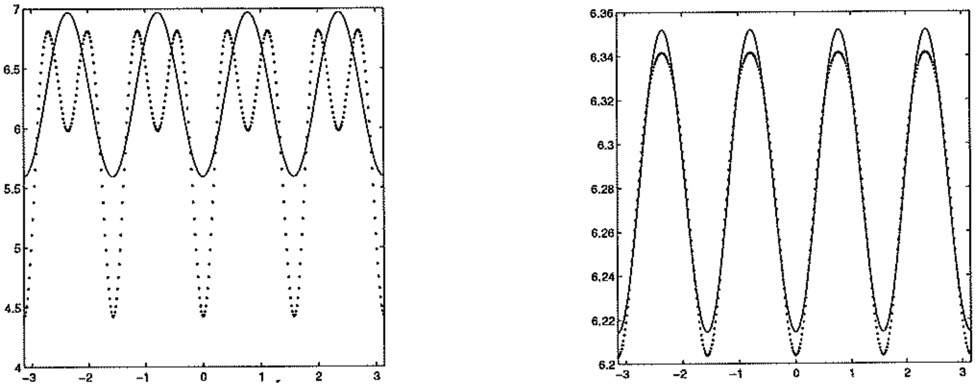

Figure 7.2: velocities due to Y (solid) and ρ (dotted), $D = 20$, $k = 4$, $\epsilon = 0.1$ (left), $\epsilon = 0.01$ (right).

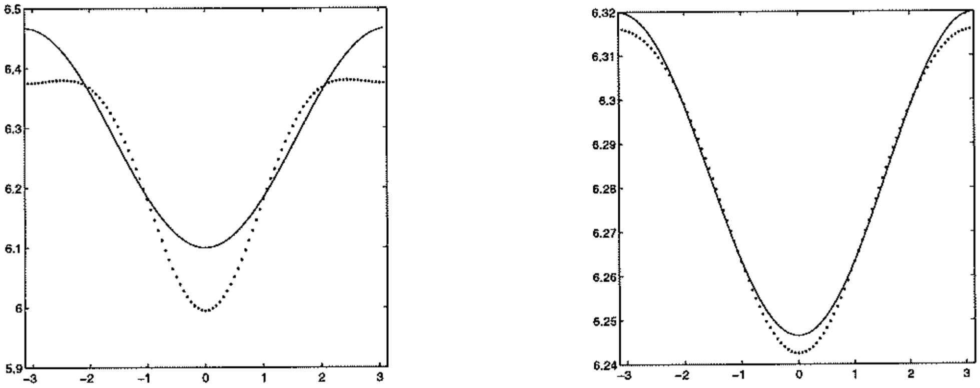

Figure 7.3: velocities due to Y (solid) and ρ (dotted), $D = 10$, $k = 1$, $\epsilon = 0.1$ (left), $\epsilon = 0.02$ (right).

However, solutions obtained using the level set method show the inherent instability and dendrites do form (see Figures 7.4 and 7.5). But these dendrites sometimes do not occur at the correct time. The computed solution for a case with a small ω may blow up faster than one with a larger ω (compare Figures 7.1 and 7.6). This is because of the appearance of small oscillations in the calculation, probably due to numerical errors, which will blow up very fast (see Figures 7.6 and 7.7) since when the frequency k is large, ω is of order k and this drives the oscillations immediately to blow-up.

7.5 Conclusion

Because of the constraint on the size of ϵ, solutions computed from the level set method may not be well resolved. Also small oscillations that may result from minute errors can quickly magnify, complicating the ability to get meaningful data. But the existence of the instability in the computational results cannot be questioned and solutions seem to grow at the correct rate when spurious oscillations do not occur.

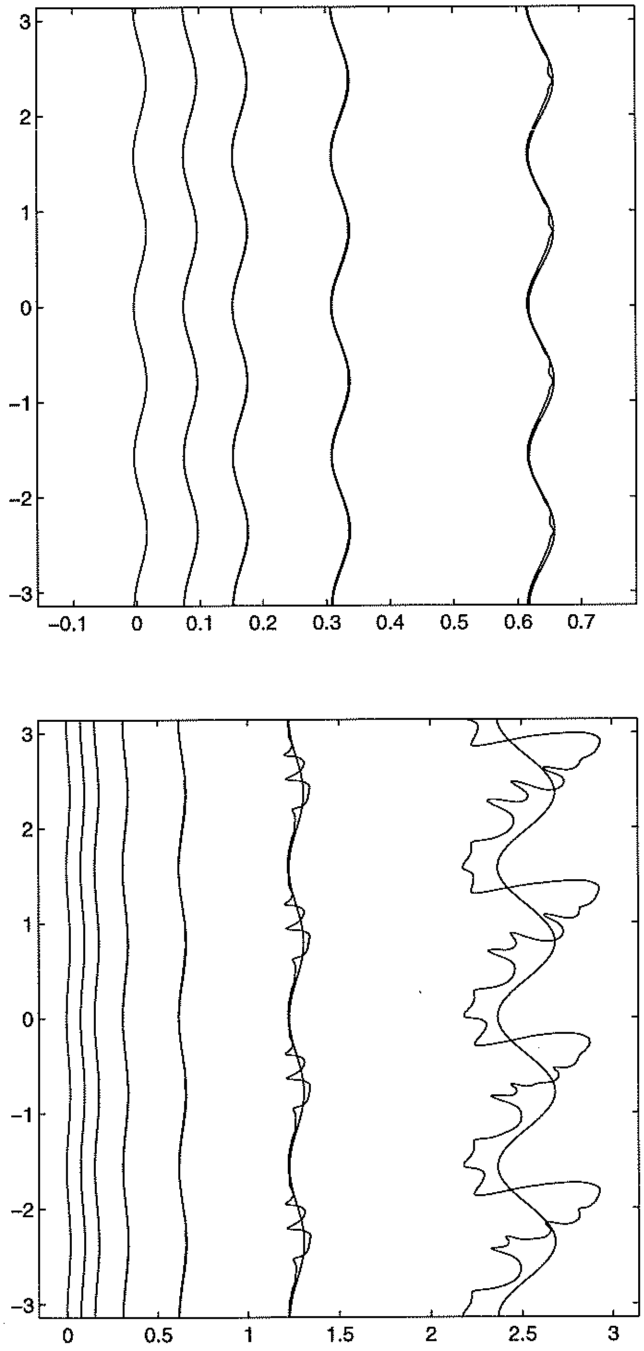

Figure 7.4: exact (smooth) and computed (dendritic) solutions, $D = 20$, $\epsilon\hat{Y}_1 = -0.01$, $k = 4$, $\omega = 6.8789$, $t = 0.0, 0.0125, 0.025, 0.05, 0.1, 0.2, 0.4$ (bottom), first 5 curves zoomed in (top).

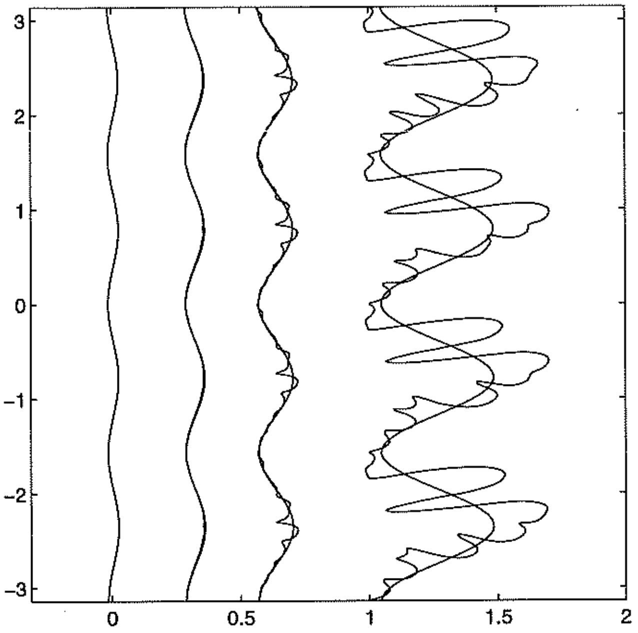

Figure 7.5: exact (smooth) and computed (dendritic) solutions, $D = 10$, $\epsilon\hat{Y}_1 = -0.02$, $k = 4$, $\omega = 11.946$, $t = 0.0, 0.05, 0.1, 0.2$.

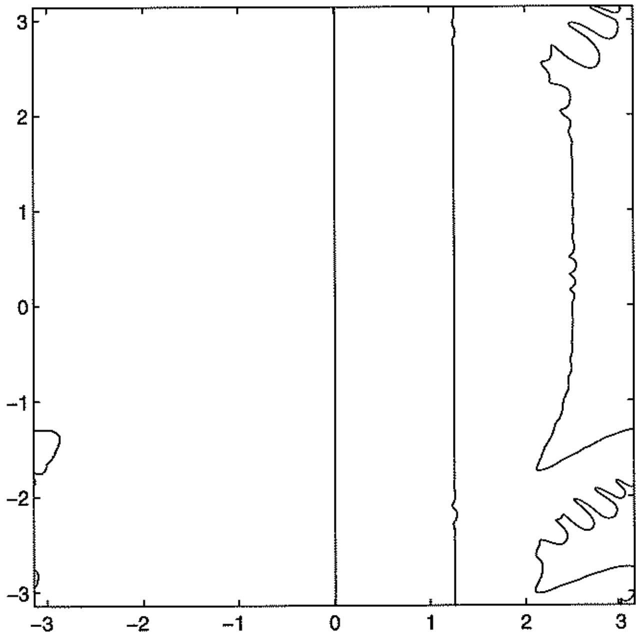

Figure 7.6: unstable computed solution of straight step, $D = 10$, $t = 0.0, 0.2, 0.4$.

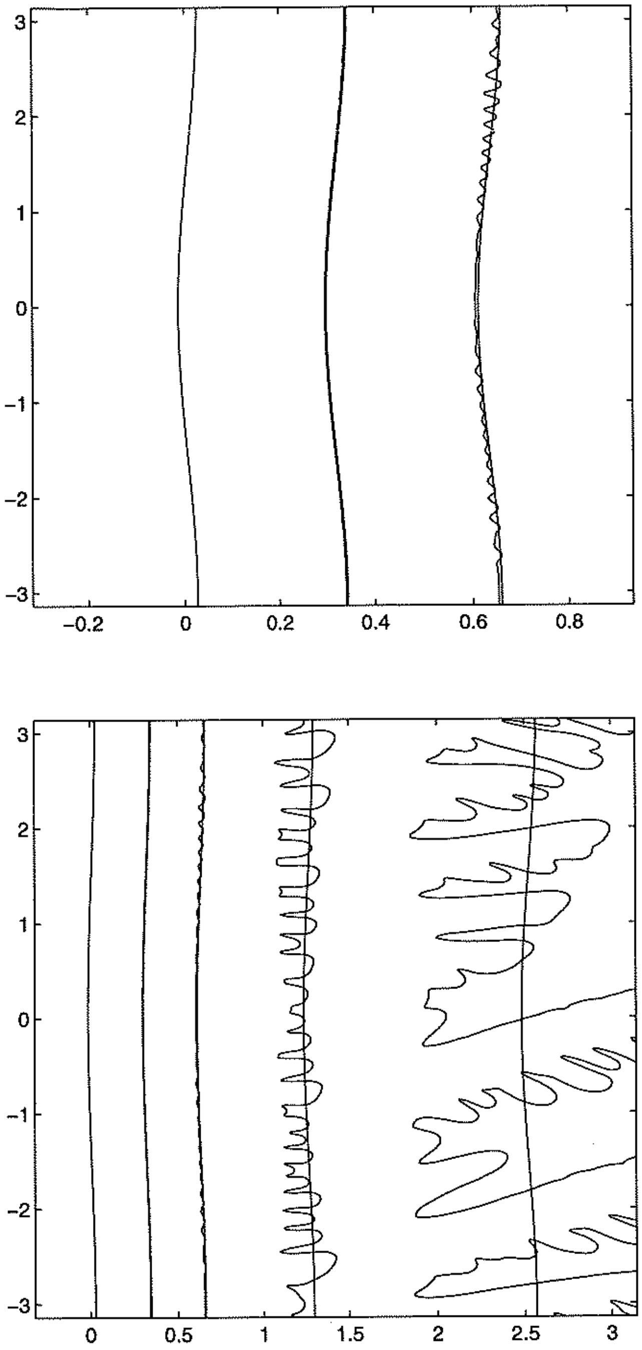

Figure 7.7: exact (smooth) and computed (dendritic) solutions, $D = 10$, $\epsilon\hat{Y}_1 = -0.02$, $k = 1$, $\omega = 1.8346$, $t = 0.0, 0.05, 0.1, 0.2, 0.4$ (bottom), first 3 curves zoomed in (top).

References

[1] S.J. Altschuler *Singularities of the curve shrinking flow for space curves.* J. Differential Geom., 34(2):491–514, 1991.

[2] L. Ambrosio and H.M. Soner. *Level set approach to mean curvature flow in arbitrary codimension.* Journal of Differential Geometry, 43:693–737, 1996.

[3] M. Bertalmio, G. Sapiro, and G. Randall. *Region Tracking on Surfaces Deforming via Level-Sets Methods.* Scale-Space, 330–338, 1999.

[4] R.E. Caflisch, M. Gyure, B. Merriman, S. Osher, C. Ratsch, D. Vvedensky, and J. Zinck. *Island Dynamics and the Level Set Method for Epitaxial Growth.* accepted for Appl. Math. Letters, 1999.

[5] T. Chan and L. Vese. *An Active Contour Model Without Edges.* Lecture Notes in Comp. Sci., 1687:141–151, 1999.

[6] S. Chen, B. Merriman, S. Osher, and P. Smereka. *A simple level set method for solving Stefan problems.* J. Comput. Phys., 135:8, 1995.

[7] S.Y. Cheng and S.T. Yau. *On the regularity of the solution of the n-dimensional Minkowski problem.* Comm. Pure Appl. Math., 29(5):495–516, 1976.

[8] D.L. Chopp. *Computing minimal surfaces via level set curvature flow.* J. Comput. Phys., 106:77–91, 1993.

[9] D.L. Chopp. *Flow Under Geodesic Curvature.* UCLA CAM Report, 92-23, 1992.

[10] K.S. Chou and X.J. Wang. *The Logarithmic Gauss Curvature Flow.* preprint, 1998.

[11] E. De Giorgi. *Barriers, boundaries, motion of manifolds.* Lectures held in Pavia, Italy, 1994.

[12] M. do Carmo. *Differential geometry of curves and surfaces.* Translated from the Portuguese. Prentice-Hall, Inc., Englewood Cliffs, N.J., 1976.

[13] T.Y. Hou, J.S. Lowengrub, and M.J. Shelley. *Removing the Stiffness from Interfacial Flows with Surface Tension.* J. Comput. Phys., 114:312, 1994.

[14] S.D. Howison, J.D. Morgan, and J.R. Ockendon. *A Class of Codimension-Two Free Boundary Problems.* SIAM Rev., 39(2):221–253, 1997.

[15] G.S. Jiang and D. Peng. *Weighted ENO Schemes for Hamilton Jacobi Equations.* UCLA CAM Report, 97-29, 1997.

[16] M. Kang. *Level Set Approach for the Motion of Soap Bubbles with Curvature Dependent Velocity or Acceleration.* UCLA CAM Report, 96-19, 1996.

[17] R. Kimmel *Intrinsic Scale Space for Images on Surfaces: The Geodesic Curvature Flow.* Graphical Models and Image Processing, 59(5):365–372, 1997.

[18] R. Kimmel and J.A. Sethian. *Computing geodesic paths on manifolds.* Proc. Natl. Acad. Sci. USA, 95:8431–8435, 1998.

[19] J. Koplik and H. Levine *Vortex Reconnection in Superfluid Helium.* Physical Review Letters, 71(9):1375-1378, 1993.

[20] L.M. Lorigo, O. Faugeras, W.E.L. Grimson, R. Keriven, R. Kikinis, and C.F. Westin. *Co-Dimension 2 Geodesic Active Contours for MRA Segmentation.* Int'l Conf. Information Processing in Medical Imaging, June/July 1999.

[21] A. Marquina and S. Osher *Explicit Algorithms for a New Time Dependent Model Based on Level Set Motion for Nonlinear Deblurring and Noise Removal.* UCLA CAM Report, 99-5, 1999.

[22] S. Osher and R. Fedkiw. *Level Set Methods.* UCLA CAM Report, 00-08, 2000.

[23] S. Osher and B. Merriman. *The Wulff Shape as the Asymptotic Limit of a Growing Crystalline Interface.* Asian J. Math., 1(3):560–571, 1997.

[24] 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.

[25] D. Peng, B. Merriman, S. Osher, H.K. Zhao, M. Kang. *A PDE-based fast local level set method.* J. Comput. Phys., 155(2):410–438, 1999.

[26] L. Rudin, S. Osher, and E. Fatemi. *Nonlinear total variation based noise removal algorithms.* Physica D, 60:259–268, 1992.

[27] S. Ruuth, B. Merriman, J. Xin, and S. Osher. *Diffusion-Generated Motion by Mean Curvature for Filaments.* UCLA CAM Report, 98-47, 1998.

[28] J.A. Sethian. *Level Set Methods.* Cambridge University Press, 1996.

[29] C.W. Shu and S. Osher *Efficient implementation of essentially nonoscillatory shock-capturing schemes.* J. Comput. Phys., 77(2):439–471, 1988.

[30] M. Sussman, P. Smerka, and S. Osher. *A level set method for computing solutions to incompressible two-phase flow.* J. Comput. Phys., 114:146–159, 1994.

[31] J.E. Taylor. *Motion by crystalline curvature.* In J.E. Taylor, editor, Computing Optimal Geometries, 63–65. AMS selected lectures in math., 1991.

[32] J.N. Tsitsiklis. *Efficient Algorithms for Globally Optimal Trajectories.* IEEE Transactions on Automatic Control, 40:1528–1538, 1995.

[33] V.E. Zakharov, S.L. Musher, and A.M. Rubenchik. *Hamiltonian Approach to the Description of Non-Linear Plasma Phenomena.* Review Section of Physics Letters, 129(5):285–366, 1985.

[34] H.K. Zhao, T.F. Chan, B. Merriman, and S. Osher. *A variational level set approach to multiphase motion.* J. Comput. Phys., 127:179–195, 1996.

[35] H.K. Zhao, S. Osher, B. Merriman, and M. Kang. *Geometric interpolation of curves and surfaces.* UCLA CAM Report, 98-7, 1998.