EFFICIENT ALGORITHMS FOR DIFFUSION-GENERATED MOTION BY MEAN CURVATURE

By

Steven J. Ruuth

BMATH, University of Waterloo, 1991

MSc, University of British Columbia, 1993

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

in

THE FACULTY OF GRADUATE STUDIES

DEPARTMENT OF MATHEMATICS

AND

INSTITUTE OF APPLIED MATHEMATICS

We accept this thesis as conforming

to the required standard

....

....

....

THE UNIVERSITY OF BRITISH COLUMBIA

August 1996

© Steven J. Ruuth, 1996

In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Department of Mathematics
The University of British Columbia
2075 Wesbrook Place
Vancouver, Canada
V6T 1W5

Date:

Abstract

This thesis considers the problem of simulating the motion of evolving surfaces with a normal velocity equal to mean curvature plus a constant. Such motions arise in a variety of applications. A general method for this purpose was proposed by Merriman, Bence and Osher, and consists of alternately diffusing and sharpening the front in a certain manner. This method (referred to as the MBO-method) naturally handles complicated topological changes with junctions in several dimensions. However, the usual finite difference discretization of the method is often exceedingly slow when accurate results are sought, especially in three spatial dimensions.

We propose a new, spectral discretization of the MBO-method which obtains greatly improved efficiency over the usual finite difference approach. These efficiency gains are obtained, in part, through the use of a quadrature-based refinement technique, by integrating Fourier modes exactly, and by neglecting the contribution of rapidly decaying solution transients. The resulting method provides a practical tool, not available hitherto, for accurately treating the motion by mean curvature of complicated surfaces with junctions. Indeed, we present numerical studies which demonstrate that the new algorithm is often more than 1000 times faster than the usual finite difference discretization.

New analytic and experimental results are also developed to explain important properties of the MBO-method such as the order of the approximation error. Extrapolated algorithms, not possible when using the usual finite difference discretization, are proposed and demonstrated to achieve more accurate results.

We apply our new, spectral method to simulate the motion of a number of three dimensional surfaces with junctions, and we visualize the results. We also propose and study a simple extension of our method to a nonlocal curvature model which is impractical to treat using the previously available finite difference discretization.

Table of Contents

	ii
List of Figures	**vi**
Acknowledgements	**ix**
1 Introduction	**1**
1.1 Curvature-Dependent Motion	1
1.2 Methods for Curvature-Dependent Motion	4
1.3 Overview	8
2 Diffusion-Generated Motion by Mean Curvature Algorithm	**10**
2.1 The Two Phase Problem	10
2.2 Multiple Junctions	13
2.3 Selection of τ	16
2.4 Finite Difference Discretizations of the MBO-Method	17
2.4.1 Selection of a Time-Stepping Method	17
2.4.2 Limitations of Finite Difference Discretizations	18
3 A New, Spectral Method	**24**
3.1 Discretization of the Heat Equation	24
3.2 Calculation of the Fourier Coefficients	26
3.3 Approximation of the Finest Subregions	30
3.3.1 Trivial Treatment of the Finest Subregions	30

3.3.2 Piecewise Linear Approximation for Two-Phase Problems 31
3.3.3 Piecewise Linear Approximations for Junctions 39
3.4 Refinement Techniques 43
3.4.1 The Original Refinement Algorithm 43
3.4.2 A Method for a Gradual Refinement 47
3.5 Fast, Transform-Based Algorithms 49
3.5.1 Overview 52
3.5.2 The Unequally Spaced Fast Fourier Transform 53
3.6 Comparison to the Usual Finite Difference Discretization 57

4 Theoretical and Numerical Studies 60

4.1 Smooth Interfaces 60
4.1.1 Truncation Error Analysis 61
4.1.2 Extrapolation 63
4.1.3 Numerical Experiments 64
4.2 Nonsmooth Boundaries 66
4.3 Singularities in the Solution as Regions Disappear 69
4.4 Junctions in Two Dimensions 70
4.4.1 Error Analysis 71
4.4.2 Numerical Experiments 77
4.5 Summary 80

5 Numerical Experiments and Visualization 82

5.1 Three Dimensional, Two-Phase Problems 82
5.1.1 Visualization 82
5.1.2 Numerical Experiments 86
5.2 Junctions in Three Dimensions 87

5.2.1 Visualization 87
5.2.2 Numerical Experiments 90
5.3 A Nonlocal Model 91

6 Conclusions **97**
6.1 Summary 97
6.2 Future Research Directions 99

Bibliography **101**

A An Estimate for the Number of Basis Functions **107**

List of Figures

1.1 A Region Evolving According to Mean Curvature Motion 2

1.2 Evolution of Multiple Grains 3

1.3 A 3-state Potts Model 6

2.1 Initial Motion 12

2.2 Characteristic Set 12

2.3 After a Time, τ 13

2.4 Sharpened Region 13

2.5 Initial Regions 15

2.6 Characteristic Sets 15

2.7 After a Time τ 15

2.8 Sharpened Regions 15

2.9 Final Regions 16

2.10 Motion by Mean Curvature Results for Crank-Nicolson at Times, t . . . 19

2.11 A Smooth Shape with Widely Varying Local Curvatures 22

2.12 A Banded, Finite Difference Mesh 22

2.13 Sharpening a Shape 23

2.14 Sharpening a Perturbed Shape 23

3.1 Subdividing the Domain into its Coarsest Subregions 28

3.2 Dividing a Subregion 29

3.3 Subdividing a Square into Triangles 33

3.4 A Shape Represented by a Triangle 33

3.5 A Shape Represented by Triangles 34
3.6 Errors Approximating Curved Segments 35
3.7 The Interpolation Step 36
3.8 Subdividing a Cube into Tetrahedrons 39
3.9 A Shape Approximated by a Tetrahedron 40
3.10 A Shape Approximated by Tetrahedrons 40
3.11 A Shape Represented as a Difference 41
3.12 Junction for which all Corner Phases Differ 42
3.13 Two Phases Represented at Corners 43
3.14 Refinement of a Smooth Region 45
3.15 Original Refinement can miss Slivers of the Region 45
3.16 A Neglected Section 46
3.17 A Problem with Sharp Corners 46
3.18 Refinement Methods for Corners 47
3.19 Gradual Refinement Captures the Entire Region 50
3.20 Refining a Cell 50
3.21 A Smooth Interface at Time, t 59

4.1 The Initial Interface 61
4.2 Extrapolated, Semi-Discrete Result 66
4.3 Initially Nonsmooth Interface at Times, t 67
4.4 The Initial Junction 72
4.5 A Smooth Three-Phase Problem 77
4.6 Evolution of a Junction Through a Singularity 79

5.1 From Q the Entire Curve is Visible 84
5.2 A Matrix Representation of the Surface 84

5.3 Splitting a Shape into Easily Parameterized Portions 85
5.4 Piecewise Constant and Gouraud Shading of the Surface 85
5.5 Thin-Stemmed Barbell Moving by Mean Curvature Motion 88
5.6 Thick-Stemmed Barbell Moving by Mean Curvature Motion 89
5.7 Composition of a Junction 90
5.8 Three Phase Example Moving by Mean Curvature Motion 92
5.9 Four Phase Example Moving by Mean Curvature Motion 93
5.10 A Test Problem for the Nonlocal Curvature Algorithm 95
5.11 Nonlocal Model Which Preserves Area 96

A.1 Contributing Rectangles 110

Acknowledgements

First and foremost, many thanks to my supervisors Dr. Uri Ascher and Dr. Brian Wetton for their many helpful suggestions while working on this thesis.

I would also like to thank Dr. Leslie Greengard for suggesting the use of the unequally spaced fast Fourier transform and NSERC for financially supporting me during my graduate work.

Chapter 1

Introduction

1.1 Curvature-Dependent Motion

The topic of curve and surface evolution has recently generated tremendous interest in the mathematical sciences community and in several areas of application such as in grain growth and image enhancement. The computation of such motion and its rigorous justification have proved to be difficult tasks. This thesis considers the numerical approximation of curvature-dependent motion of surfaces with multiple phases. In particular, we study and develop fast methods for the case where the normal velocity, v_n, of a surface is given by the sum of its principal curvatures (or mean curvature), κ,

$$v_n(\vec{x}, t) = \kappa(\vec{x}, t). \tag{1.1}$$

We shall refer to this type of motion as motion by mean curvature. As illustrated for the two dimensional region of Figure 1.1, such a motion causes the most highly curved portions of a curve to smooth most rapidly. Indeed, any simple, closed curve in the plane shrinks to a small circle and disappears, regardless of the initial shape [31, 36]. For higher dimensional shapes, the most curved portions also move most quickly; however, topological merging and breaking is also possible [20].

Certain idealized models for grain growth fit into the framework of mean curvature motion. For example, when a liquid metal solidifies, crystallization begins in many locations with random orientations. As crystals grow, grains with different orientations meet, to form interfaces. To a good approximation, the surface energy of such a material

Initial Motion *Later Region (Solid)*

Figure 1.1: A Region Evolving According to Mean Curvature Motion

is isotropic or independent of the orientation of the boundary of each grain [9]. By annealing the metal, it becomes warmed so that boundary atoms can change their phase. This produces an interface motion proportional to mean curvature motion [48, 51, 9]. In the interesting case where three or more grain orientations are present, junctions of moving surfaces can occur. This is illustrated in Figure 1.2 for the case of three grains. Although these important problems have been the subject of some study in two dimensions (e.g., [13, 14]), little is known about their numerical solution, especially in three dimensions.

An extension of the mean curvature model that we shall consider arises when the thermodynamic driving force of the interface motion depends on the volume phase change (i.e. bulk effects) as well as surface effects [71]. In this case, the normal velocity of the surface is given by its mean curvature plus a constant,

$$v_n(\vec{x}, t) = \kappa(\vec{x}, t) + c(t). \tag{1.2}$$

We shall refer to this type of motion as affine velocity front motion.

Curve evolution has also been studied extensively for image analysis and the enhancement of planar shape [43, 50, 42, 67, 41, 72]. Important application areas for this subject include noise suppression, image recognition and image interpretation. Mean curvature

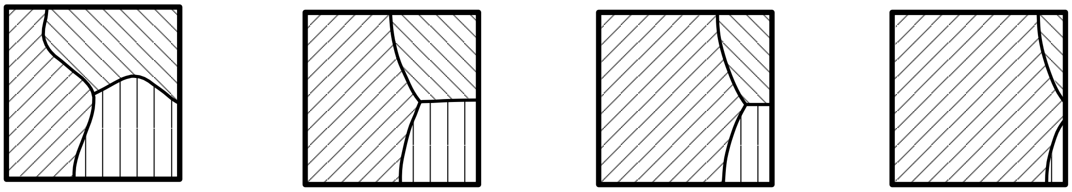


Figure 1.2: Evolution of Multiple Grains

motion and affine motion are among the many curve evolution models that have been proposed for these applications. For example, affine velocity motion has been used to produce topological and other shape changes for characterizing shapes [42]. For other applications of these motions, see [42, 67] and references therein.

Another important model that we will be considering fits into the framework of affine motion (1.2) and occurs when

$$c(t) = -\kappa_{av}(t)$$

where $\kappa_{av}(t)$ is the average mean curvature over the surface at time t. Such motion preserves phase volumes (or areas in two dimensions) and occurs as a limit of a nonlocal model for binary alloys [57]. In the context of image enhancement, this nonlocal motion has also been suggested as a possible smoothing which preserves the area of shapes [64].

Theoretical aspects of mean curvature motion have also been studied extensively (see, e.g., [31, 36]). Some areas of computational interest include the approximation of bifurcation values [53] and the determination of self-similar solutions [21] under curvature flow. Computations of minimal surfaces have also been carried out by evolving surfaces according to mean curvature motion [20, 74]. Such surfaces find application in numerous areas [20] including soap film shapes, relativity theory, medical technology and architecture.

Other applications for affine motion occur in certain flame propagation problems (e.g., grassfire flow) and automatic grid generation. See [54, 63, 66] for further details.

1.2 Methods for Curvature-Dependent Motion

To study the phenomena outlined in the previous section, several numerical methods have been developed. Most of these can be divided into one of two groups: direct, or front tracking methods, where the motion of the interface is explicitly considered; and indirect methods, where the interface is given implicitly as the level set of some function.

Several direct or front tracking methods have been proposed. In [14], a direct discretization of the evolution equation for each interface is used to produce curvature-dependent motion for junctions in two dimensions. A number of other methods based on heuristic arguments have also been proposed for two dimensional junctions (see, e.g., [29, 30]). These methods are typically quite efficient for curves that never cross because they explicitly approximate the motion of the interface rather than a level set of some higher dimensional function. When line or planar segments interact, however, decisions must be made as to whether to insert or delete segments. Because very complicated topological changes can occur in three dimensions, implementation of front tracking methods is often impractical in more than two dimensions.

Another direct method, Brakke's method [9], minimizes surface energy to produce a variety of motions including motion by mean curvature. For three dimensional problems, however, this method requires user intervention when topological changes occur [9]. Also, there is no proof that the method actually approximates motion by mean curvature [71], although it is known that the Allen-Cahn equation (see below) yields Brakke's motion in the limit for two-phase problems [40].

Phase field methods (see [16, 71] and references therein) give the interface as a level set of a reaction-diffusion equation such as the Allen-Cahn equation,

$$u_t = \epsilon \Delta u - \frac{1}{\epsilon} f'(u).$$

Here f' is the derivative of a double well potential (e.g., $f'(u) = u(u-1)(u-\frac{1}{2})$) and ϵ is a small parameter. For the two-phase problem, it has been proven that certain phase field models produce mean curvature motion when $\epsilon \to 0$ [59, 58, 12, 26]. (Indeed, an important motivation for studying mean curvature motion is that it arises as a limit of certain phase field models.) By letting ϵ be proportional to $\frac{\nabla u}{|\nabla u|}$, these methods provide a means of handling anisotropy (albeit in an ad hoc fashion) [71]. Junctions have also been treated by considering a vector-valued u [13, 56]. Unfortunately, phase field methods are often inherently too expensive for practical computation [48] because they represent the interface as an internal layer and thus require an extremely fine mesh (at least locally) to resolve this layer.

Monte-Carlo methods for Q-state Potts models have also been used to simulate curvature-dependent motion with junctions (see, e.g., [33]). The Q-state Potts model assigns an integer state between 1 and Q to each point on a lattice to form regions (see Figure 1.3). The energy of each lattice site is taken to be a weighted average of the number of neighboring sites which are at a different state. To evolve regions, a random site and state are chosen. The selected site is set to the new state with a probability depending on temperature if the energy increases, and with probability 1 otherwise. It is unclear, however, what type of continuous motion this stochastic model approximates [71]. This method also introduces unwanted anisotropy into the motion due to the spatial mesh [71]. Furthermore, our numerical experiments indicate that these statistical methods are typically too slow to find accurate approximations to mean curvature motion.

The Hamilton-Jacobi level set method of Osher and Sethian [54] is an appropriate

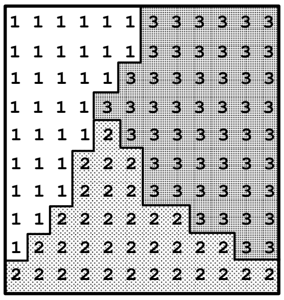


Figure 1.3: A 3-state Potts Model

choice for a wide variety of two-phase problems. This method naturally handles topological merging and breaking in any number of spatial dimensions. To carry out the Osher-Sethian method for an initial surface Γ^0, a continuous function

$$\Phi^0 : \Re^n \to \Re$$

is selected such that

$$\Gamma^0 = \{x \in \Re^n : \Phi^0(x) = 0\}.$$

By solving

$$\begin{aligned} \Phi_t &= F(\kappa)|\nabla\Phi| \\ \Phi(x,0) &= \Phi^0(x) \end{aligned}$$

for an arbitrary function of curvature, $F(\kappa)$, we obtain a surface

$$\Gamma(t) = \{x : \Phi(x,t) = 0\}$$

which moves with a normal velocity $F(\kappa)$. It has been rigorously proven that this method converges for the case of mean curvature motion [27]. Furthermore, by assigning a function, Φ_i, to each region and evolving according to the method an extension to problems with junctions is possible [48]. This coupled Osher-Sethian approach allows for the specification of arbitrary velocities on each front. Unfortunately, this method can be excessively slow in three dimensions, in part because each Φ_i must be set to the signed distance to the interface at each step of the method.

A very recent variational level set approach [74] has also been proposed for treating affine velocity motion. This method uses the level set formulation of Osher and Sethian [54] to arrive at an algorithm for a theoretical variational problem posed in [56]. This approach is reported to treat problems exhibiting topological changes with junctions in three dimensions [74]. However, this method cannot currently treat certain problems involving more than three phases. Furthermore, the usefulness of the method is currently limited by a time-stepping restriction similar to that associated with an explicit treatment of the heat equation. Such a restriction can produce an inefficient method for mean curvature flows because very small time steps must be used whenever a fine spatial mesh is applied.

A method based on the model of diffusion-dependent motion of level sets has recently been proposed by Merriman, Bence and Osher [47, 48]. We shall refer to this method as the MBO-method (cf. [25]) although the name DGCDM-algorithm has also been used [48]. The specifics of this method are elaborated upon in subsequent chapters, so will not be repeated here. This method naturally handles complicated topological changes with junctions in several dimensions. It also produces accurate approximations to mean curvature flow more efficiently than phase field models, the recent variational approach or Monte-Carlo methods. Thus, this method is often the only practical choice for three dimensional problems involving junctions. Even so, the method is often exceedingly slow for three dimensional and affine velocity motions. Similarly to all other methods for multiple-phase problems, no convergence results are known for the MBO-method. However, [25, 4] do give rigorous convergence proofs for two-phase mean curvature motion and [45] gives some further asymptotic results.

In this thesis, we describe a fast, new algorithm which improves upon the speed of the usual discretization of the MBO-method, often by a factor of a thousand or more. We also develop analytic and experimental results to explain important properties of the method, such as the order of the approximation error and its enhancement by extrapolation. Finally, the improved utility of the new algorithm is demonstrated by approximating a number of two and three dimensional motions including a nonlocal curvature model. The resulting method provides a practical tool, not available hitherto, for accurately approximating the motion by mean curvature of complicated surfaces with junctions.

1.3 Overview

An outline of the rest of the thesis follows.

In Chapter 2, algorithms describing the MBO-method for two phase and multiple phase problems are given. This is followed by a discussion on how to select the step size, τ, of the method. For the case of the finite difference discretizations originally proposed [47], the selection of an appropriate time-stepping scheme is discussed and several limitations of the method are identified.

In Chapter 3, a new, spectral method for the realization of the MBO-method is proposed and described in detail. A spatial discretization is given and an efficient quadrature for calculating the corresponding Fourier coefficients is provided. This quadrature obtains accurate approximations to the front using a piecewise linear approximation to the surface and a gradual refinement technique. Unequally spaced transform methods for the rapid evaluation of the Fourier coefficients are also applied. This chapter concludes with a comparison of the proposed method and the usual finite difference approach. In particular, numerical experiments are presented to illustrate the efficiency gains which arise from our method.

In Chapter 4, the order of accuracy of the MBO-method is studied for a variety of two dimensional shapes which move according to mean curvature motion. At first, local error expansions and numerical studies are carried out to demonstrate that the method produces a first order error in τ for smooth, two-phase problems. Next, numerical studies for nonsmooth corners and for singularities in the solution are discussed. The error here appears to be $\mathcal{O}(\tau \log(\tau))$. For three-phase problems, an expansion is derived which suggests that junction angles are stable and remain within $\mathcal{O}(\sqrt{\tau})$ of the correct configuration. Numerical studies are also given which confirm that an $\mathcal{O}(\sqrt{\tau})$ error arises for junctions. Throughout this chapter, extrapolated algorithms are proposed and demonstrated to achieve more accurate results. While the discussion in Chapter 4 is independent of the discretization method used to implement the MBO-method, the spectral method of Chapter 3 was found crucial to realize the claimed errors at an affordable cost.

In Chapter 5, the new method for discretizing the MBO-method is applied to a number of three dimensional surfaces. In particular, two-phase barbell-shaped regions are evolved according to mean curvature motion and the results are visualized. Multiple phase motions are also approximated for examples involving three and four-phase junctions in three dimensions. A simple extension of the MBO-method to a nonlocal curvature model is also proposed and studied.

Overall, the results are rather encouraging. Conclusions and suggestions of directions for future research are presented in Chapter 6.

Chapter 2

Diffusion-Generated Motion by Mean Curvature Algorithm

An algorithm for following interfaces propagating according to mean curvature motion (1.1) was introduced in a paper by Merriman, Bence and Osher [47, 48]. This algorithm is one of the only methods of treating both topological changes and junctions. This chapter describes the method for the two phase and multiple phase problems and discusses some serious limitations of the usual finite difference discretization of the method. Subsequent chapters discuss a new method which greatly improves upon the usual finite difference approach.

In this and later chapters, we will make use of a simplified version of the Von Neumann-Mullins parabolic law [51]. Specifically, we will use the fact that the area, A, enclosed by a simple curve which moves by mean curvature motion obeys

$$\frac{\mathrm{d}A}{\mathrm{d}t} = -2\pi. \tag{2.3}$$

2.1 The Two Phase Problem

Suppose we wish to follow an interface moving by mean curvature motion (see, e.g., Figure 2.1). To carry out this motion over a domain, $\mathcal{D}$, the MBO-method uses a diffusion-generated motion:

MBO-Method (Two Regions)

BEGIN

(1) Set U equal to the characteristic function for the initial region.

i.e., set $U(\vec{x},0) = \begin{cases} 1 & \text{if } \vec{x} \text{ belongs to the initial region} \\ 0 & \text{otherwise.} \end{cases}$

REPEAT for all steps, j, from 1 to the final step:

BEGIN

(2) Apply diffusion to U for some time, τ.

i.e., find $U(\vec{x}, j\tau)$ using $\begin{cases} U_t &= \Delta U, \\ \frac{\partial U}{\partial n} &= 0 \text{ on } \partial\mathcal{D} \end{cases}$

starting from $U(\vec{x},(j-1)\tau)$.

(3) "Sharpen" the diffused region by setting

$U(\vec{x}, j\tau) = \begin{cases} 1 & \text{if } U(\vec{x}, j\tau) > \frac{1}{2} \\ 0 & \text{otherwise.} \end{cases}$

END

END

For any time t, the level set $\{\vec{x} : U(\vec{x},t) = \frac{1}{2}\}$ gives the location of the interface.

In step (2) of the method, zero flux conditions

$$\frac{\partial U}{\partial n} = 0 \text{ on } \partial\mathcal{D} \tag{2.4}$$

are selected. These boundary conditions cause the curve to meet the boundary at right angles, as is appropriate for the case of grain growth [14]. It has also been argued that these boundary conditions (2.4) are the most natural for image processing since they do not impose any value at the boundary [2]. Although our experiments will concentrate on the important zero flux case, other types of boundary conditions are sometimes considered. In particular, Dirichlet conditions have been used for computing minimal surfaces [20].

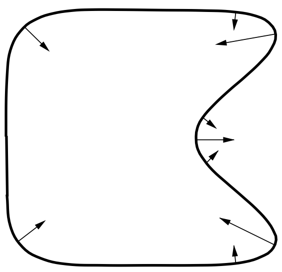


Figure 2.1: Initial Motion

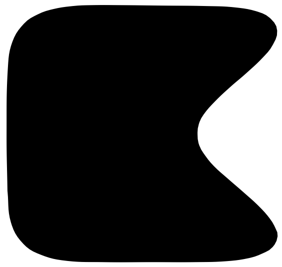


Figure 2.2: Characteristic Set

To illustrate this algorithm for the problem of Figure 2.1, a grey-scaled representation of U after each of the steps (1) to (3) is given below:

After step (1) $U = 1$ for the black region of Figure 2.2 and $U = 0$ elsewhere,

(2) U ranges between 0 and 1 as represented by the greyscale image[1] of Figure 2.3,

(3) $U = 1$ for the black region of Figure 2.4 and $U = 0$ elsewhere.

An extension to the case of affine velocity front motion (1.2) is also possible. To obtain such a motion, we track the level set

$$\frac{1}{2} - \frac{1}{2}c(t)\sqrt{\frac{\tau}{\pi}}$$

instead of the usual level set of $\frac{1}{2}$ [45].

[1]The "ringing" in Figure 2.3 and others is an artifact which arises from printing. These features are not present in the original grayscale images.

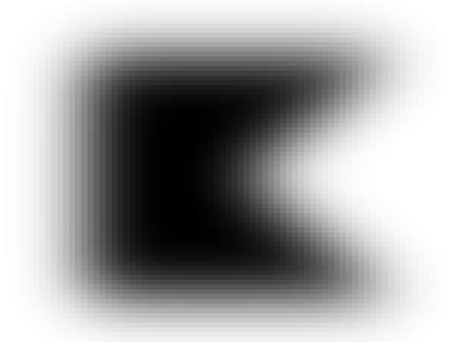


Figure 2.3: After a Time, τ

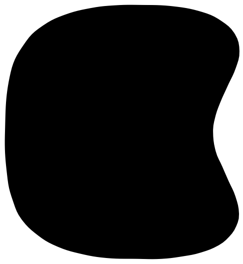


Figure 2.4: Sharpened Region

2.2 Multiple Junctions

An extension of the previous method to multiple phases has also been made [48]. To treat such problems, the characteristic function for each region is diffused for a time τ, as is outlined below for the case of 120-120-120 degree junction angles[2].

MBO-Method (Multiple (r) Regions)

BEGIN

(1) For $i = 1, \ldots, r$

Set $U_i(\vec{x}, 0)$ equal to the characteristic function for the ith region.

REPEAT for all steps, j, from 1 to the final step:

BEGIN

[2]See [45, 48] for an extension to nonsymmetric junction angles.

(2) For $i = 1, \ldots, r$, starting from $U_i(\vec{x}, (j-1)\tau)$,

Apply diffusion to U_i for some time slice, τ.

i.e., find $U_i(\vec{x}, j\tau)$ using $\begin{cases} \frac{\partial U_i}{\partial t} &= \Delta U_i, \\ \frac{\partial U_i}{\partial n} &= 0 \text{ on } \partial \mathcal{D}. \end{cases}$

(3) "Sharpen" the diffused regions by setting the largest U_i equal to 1 and the others equal to 0 for each point on the domain:

For $i = 1, \ldots, r$

Set $U_i(\vec{x}, j\tau) = \begin{cases} 1 \text{ if } \vec{x} \in \{\vec{y} \in \mathcal{D} : U_i(\vec{y}, j\tau) \geq U_j(\vec{y}, j\tau), j = 1, \ldots, r\} \\ 0 \text{ otherwise} \end{cases}$

END

END

For any time t, the interfaces are given by

$$\bigcup_{i=1,\ldots,r} \{\vec{x} : U_i(\vec{x}, t) = \max_{j \neq i} \{U_j(\vec{x}, t)\}\}. \tag{2.5}$$

To illustrate this algorithm for the problem of Figure 2.5, a grey-scaled representation of U_i after each of the steps (1) to (3) is given below:

After step (1) $U_i = 1$ for the black regions of Figure 2.6 and $U_i = 0$ elsewhere,

(2) U_i ranges between 0 and 1 as represented by the greyscale images of Figure 2.7,

(3) $U_i = 1$ for the black regions of Figure 2.8 and $U_i = 0$ elsewhere.

Reconstruction of the interfaces by (2.5) then gives the final regions displayed in Figure 2.9.

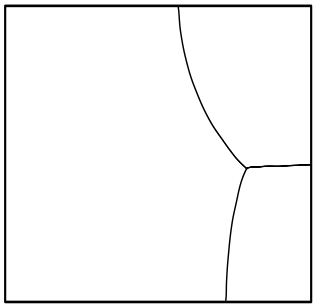


Figure 2.5: Initial Regions

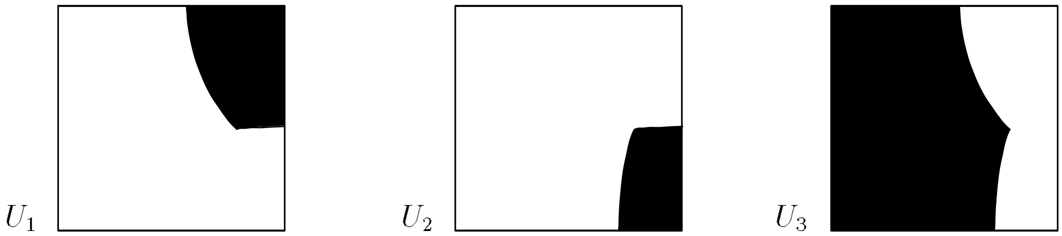


Figure 2.6: Characteristic Sets

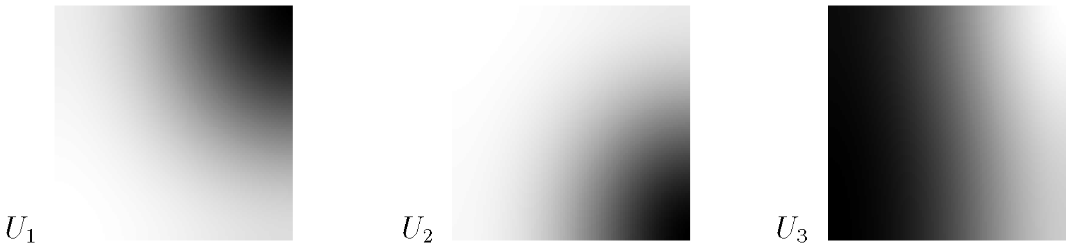


Figure 2.7: After a Time τ

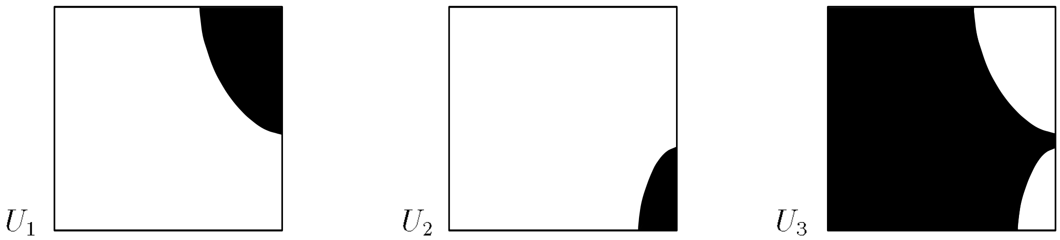


Figure 2.8: Sharpened Regions

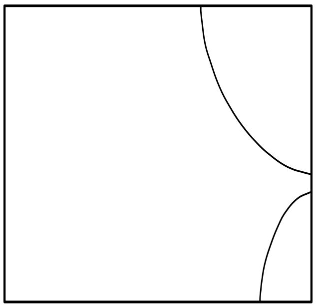


Figure 2.9: Final Regions

2.3 Selection of τ

To accurately resolve the motion of features of the interface, it is also important to select τ appropriately. In particular, τ should be small enough so that diffusive information does not travel a distance comparable to the local radius of curvature, $\frac{1}{\kappa}$ (see [48]). By Equation (2.3), it is straightforward to show that a shrinking circular interface of curvature, κ, disappears at time $t = \frac{1}{2\kappa^2}$. This gives a restriction on τ,

$$\sqrt{\tau} \ll \frac{1}{\kappa}. \tag{2.6}$$

Diffusion must also proceed long enough so that the motion of the interface over each step can be resolved by the spatial discretization. For the case of a finite difference discretization, the level set $U = \frac{1}{2}$ must move at least one grid point, otherwise the interface remains stationary. This produces the restriction that

$$\text{(speed of motion of the interface)} \times \tau \gg \text{grid spacing}$$

Letting κ be the curvature and h the grid spacing, we arrive at a second restriction for the finite difference approach,

$$\kappa\tau \gg h. \tag{2.7}$$

As we shall see, the restriction (2.7) does not appear for the new, spectral method that we propose in Chapter 3.

2.4 Finite Difference Discretizations of the MBO-Method

We now discuss several aspects of the usual finite difference approach for simulating the MBO-method. In particular, this section considers how to select a time-stepping scheme and outlines several limitations of finite difference discretizations of the MBO-method.

2.4.1 Selection of a Time-Stepping Method

Explicit time-stepping for the diffusive step of the MBO-method is very expensive because of the usual stability time step restriction arising from the heat equation. This stability time step restriction can be overcome by using implicit time-stepping schemes.

When applied to the semi-discrete heat equation,

$$\dot{\vec{U}} = \Delta_h \vec{U}$$

some implicit methods, such as backward Euler,

$$\frac{\vec{U}^{n+1} - \vec{U}^n}{\Delta t} = \Delta_h \vec{U}^{n+1}$$

give a strong decay of high frequency error modes. Other methods, such as Crank-Nicolson,

$$\frac{\vec{U}^{n+1} - \vec{U}^n}{\Delta t} = \Delta_h \left(\frac{\vec{U}^{n+1} + \vec{U}^n}{2} \right)$$

and ADI methods [69] give a very weak decay of these error modes [3, 69]. After the sharpening step of the MBO-method, the solution is discontinuous. Because high frequency modes make a very important contribution to such a result, it is essential to use a time-stepping scheme which produces appropriate damping (i.e., strong damping) of these components.

To illustrate this idea, we shall consider the motion by mean curvature of a shrinking circular interface with initial radius $\frac{1}{4}$. By Equation (2.3), the exact solution of this problem at time t is a circle of radius $\sqrt{(\frac{1}{4})^2 - 2t}$. Using a finite difference simulation of the MBO-method, this circular interface was evolved to time $t = 0.02$. The spatial discretization was carried out using the standard 5-point Laplacian with grid spacing $h = 1/256$. For the time discretization, both backward Euler and Crank-Nicolson were considered using a time step, $\Delta t = 0.0005$. The value of τ was chosen to be a multiple of Δt such that

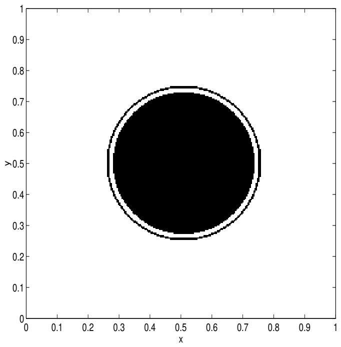
$$\tau \approx \frac{1}{10}\left(\frac{1}{16} - 2t\right).$$

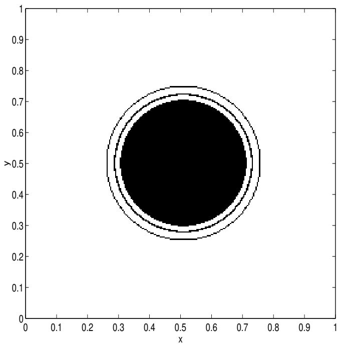
Because the solution to this problem at time t is a circle of radius $\sqrt{(\frac{1}{4})^2 - 2t}$, it is easy to show that this choice satisfies both restrictions (2.6) and (2.7). For general problems, however, an appropriate value of τ is difficult or impossible to determine using Equations (2.6) and (2.7).

In Figure 2.10, numerical results are given for Crank-Nicolson at various times, t. These results demonstrate that high frequency error modes can linger on to produce disastrous results when a weakly damping time-stepping scheme such a Crank-Nicolson is used. Application of the strongly damping backward Euler, however, gives the correct phase areas to within 3%. This simple time-stepping technique is often adequate for diffusion-generated motion by mean curvature because other, larger sources of error dominate (see Chapter 4).

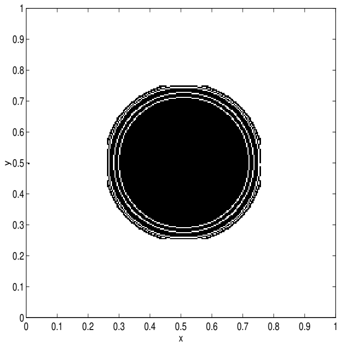
2.4.2 Limitations of Finite Difference Discretizations

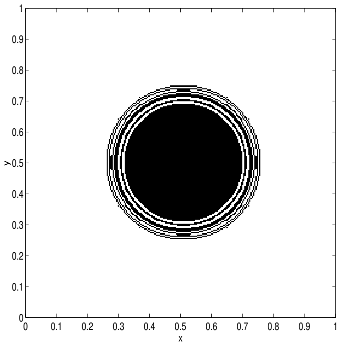
We have seen from the previous section that there are restrictions on our choice of τ and h for finite difference discretizations of the MBO-method. To produce an accurate result, τ should be small enough (see Equation (2.6)) so that diffusive information does not travel a distance on the order of the local radius of curvature. Once a sufficiently small τ is selected, the mesh spacing, h, must be chosen small enough (see Equation (2.7)) so





$t = 0.002$ $t = 0.009$





$t = 0.013$ $t = 0.020$

Figure 2.10: Motion by Mean Curvature Results for Crank-Nicolson at Times, t

that the level set moves at least one grid point, otherwise the sharpening step leaves the front stationary.

Since the MBO-method does not explicitly evaluate curvature, we would prefer to select τ and h without using Equations (2.6) and (2.7). One simple method for adaptively choosing τ for a given mesh, is to set τ to a multiple of the time taken for the level set to move one grid point. Such a choice ensures that the front propagates and produces τ-values which are roughly inversely proportional to the maximum speed of propagation of the front. Unfortunately, numerical tests indicate that an appropriate choice of the multiple is not always possible because it depends on factors such as the local curvature of the interface and the local mesh spacing. Thus, this adaptive method is often inappropriate for finding accurate results.

Satisfying the restriction (2.7) can be computationally impractical even for smooth, two dimensional problems. Consider, for example, the motion by mean curvature of the boundary of the spiral region given in Figure 2.11. (The curvature-dependent motion of similar shapes has been considered in biological models [73].) Since the local curvature of the boundary of this problem varies tremendously, it is impractical to satisfy (2.7) everywhere using a uniform mesh.

To achieve a more efficient finite difference algorithm, one might consider discretizing the MBO-method using a local mesh refinement at the level of the PDE. However, carrying out local mesh refinement is rather involved for level set methods when curvature terms arise (see [49]). An alternative approach is to place a narrow band of grid points around the front (cf. [1]). Even this optimized, finite difference approach can lead to a prohibitive number of operations per step when an accurate solution is sought.

For example, consider the motion by mean curvature of a smooth curve. For such a curve, each step of the MBO-method produces an $\mathcal{O}(\tau^2)$ error in the position of the front (see Section 4.1.1). To preserve the overall accuracy of the method, grid points must be at most a distance $\mathcal{O}(\tau^2)$ apart since each step produces an error which is comparable to the mesh spacing. Noting that the front travels a distance $\mathcal{O}(\tau)$ per step of the method (see Section 4.1.1), it is clear that a minimum of $\mathcal{O}\left(\frac{1}{\tau^3}\right)$ grid points are needed to safely band a curve (see, e.g., Figure 2.12). Thus, a minimum of $\mathcal{O}\left(\frac{1}{\tau^3}\right)$ operations per step are required to preserve the overall accuracy of the method, which is often prohibitively expensive when accurate results are sought.

A further limitation of the finite difference approach is that the error is not regular. Specifically, very small differences in the position of the level set 1/2 before sharpening can produce jumps in the front location after sharpening. This type of error is undesirable because it makes the construction of higher order accurate, extrapolated results impractical. Figures 2.13 and 2.14 illustrate how a small change in the position of the level set 1/2 can lead to a jump in the front location after sharpening. (We shall see in the next two chapters that our proposed method essentially eliminates the spatial error to allow for higher order accurate extrapolations in τ.)

To avoid the limitations outlined in this section, we introduce a new, spectral method for realizing the MBO-method in the next chapter.

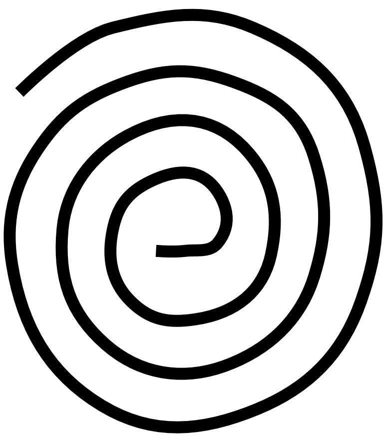


Figure 2.11: A Smooth Shape with Widely Varying Local Curvatures

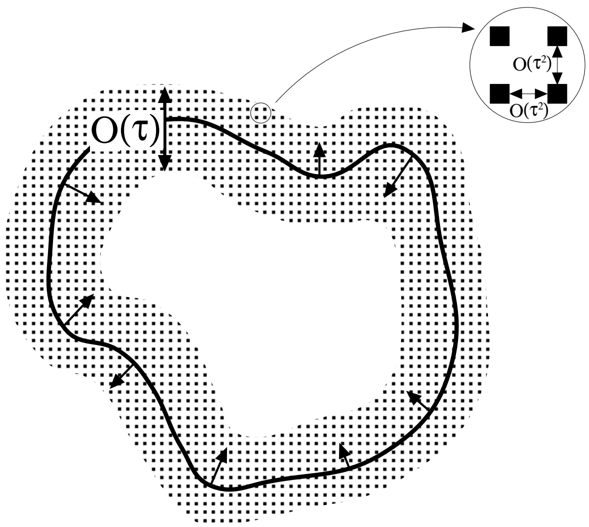


Figure 2.12: A Banded, Finite Difference Mesh

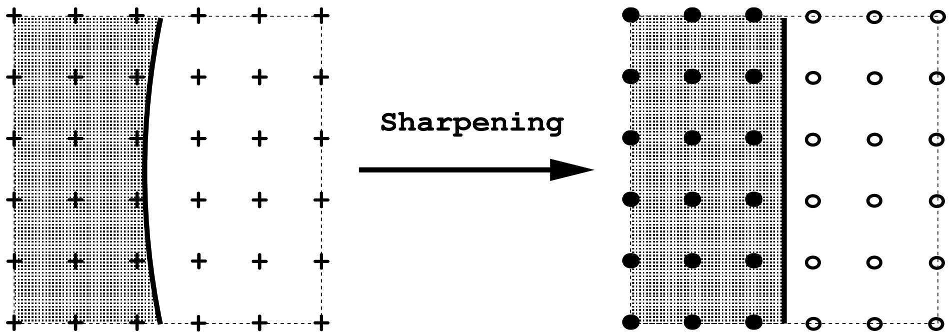


Figure 2.13: Sharpening a Shape

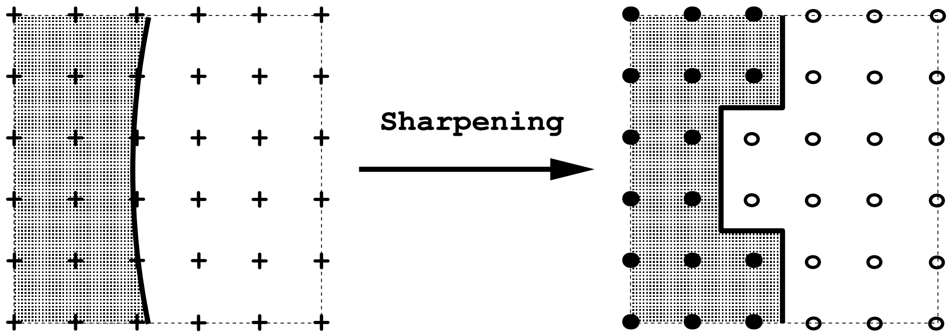


Figure 2.14: Sharpening a Perturbed Shape

Chapter 3

A New, Spectral Method

As we shall see later in this chapter, accurate computation of solutions using the usual finite difference discretization of the MBO-method can be expensive, even for simple two dimensional problems. Since we are mainly interested in three dimensional problems or problems involving more than two phases, a faster method is desired. This chapter describes a new, spectral method for realizing the MBO-method which is typically much faster than the usual finite difference approach.

For notational simplicity, the algorithm focuses on the two dimensional case over the domain, $\mathcal{D} = [0,1] \times [0,1]$. Certain extensions to three spatial dimensions and more phases are also discussed.

3.1 Discretization of the Heat Equation

As we have seen in Section 2.1, carrying out diffusion-generated motion by mean curvature requires us to solve the heat equation

$$\begin{aligned} u_t &= \Delta u, \\ \frac{\partial u}{\partial n} &= 0 \text{ on } \partial\mathcal{D} \end{aligned} \tag{3.8}$$

repeatedly over time intervals of (possibly variable) length τ, starting from the characteristic function of the region to be followed. Over any of these time intervals, u may be approximated by the Fourier cosine tensor product,

$$U(x,y,t) = \sum_{i,j=0}^{n-1} d_{ij}(t)\cos(\pi i x)\cos(\pi j y) \tag{3.9}$$

because zero flux boundary conditions are considered. Substitution of (3.9) into (3.8), then gives

$$U(x,y,t) = \sum_{i,j=0}^{n-1} c_{ij}\exp(-\pi^2[i^2+j^2][t-t_{start}])\cos(\pi i x)\cos(\pi j y) \tag{3.10}$$

for $t_{start} \le t \le t_{start} + \tau$, where $c_{ij} = d_{ij}(t_{start})$ and t_{start} is the time when the current interval starts.

One might expect that a Fourier spectral approximation for u would be unwise because u is initially discontinuous at interfaces. We are only interested in the solution after a time τ, however. After a sufficiently large time τ, high frequency modes have dissipated. Since the problem is linear, different modes do not interact (they are eigenfunctions) and thus there is never a need to approximate high frequency modes (not even near t_{start}, when high frequency modes make an important contribution to the solution). For this reason, an accurate approximation to (3.8) at time τ can be obtained using far fewer basis functions than might otherwise might be expected. Indeed, Appendix 1 shows that for smooth problems in two dimensions, selecting the number of Fourier modes in each direction to satisfy

$$n \ge \sqrt{\frac{|\ln(\frac{\epsilon}{16\pi L})|}{\pi^2\tau}}$$

for a curve of length L produces an error in the position of the front which is at most $\epsilon + \mathcal{O}(\epsilon\tau)$. In practice, however, our implementations simply select an n satisfying

$$n \ge \sqrt{\frac{|\ln(\epsilon)|}{\pi^2\tau}} \tag{3.11}$$

and verify the corresponding results by repeating the calculation with a different n (cf. [8]).

3.2 Calculation of the Fourier Coefficients

The values of the Fourier coefficients, c_{ij}, of equation (3.10) must still be determined at the beginning of each time step (i.e., immediately following the sharpening of the previous step). In fact, we carry out sharpening (at $t = t_{start}$) as part of the calculation for the Fourier coefficients. These coefficients are found using an adaptive quadrature method rather than a pseudospectral method. Begin by defining

$$R_t = \{(x, y) : U(x, y, t) > \frac{1}{2}\}$$

to be the approximation of the phase we are following. By multiplying equation (3.10) at time $t = t_{start}$ by $\cos(\pi i x)\cos(\pi j y)$ and integrating over the domain we obtain

$$\begin{aligned}\int_0^1 \int_0^1 \sum_{k,l=0}^{n-1} c_{kl} \cos(\pi k x)\cos(\pi i x)\cos(\pi l y)\cos(\pi j y)\; dx\; dy \\ = \int_0^1 \int_0^1 U(x, y, t_{start})\cos(\pi i x)\cos(\pi j y)\; dx\; dy\end{aligned}$$

which simplifies via the usual orthogonality conditions to give

$$\begin{array}{rcll} c_{00} & = & \int_0^1 \int_0^1 U(x, y, t_{start})\; dx\; dy, & \\ c_{i0} & = & 2\int_0^1 \int_0^1 U(x, y, t_{start})\cos(\pi i x)\; dx\; dy & \text{for } i \neq 0, \\ c_{0j} & = & 2\int_0^1 \int_0^1 U(x, y, t_{start})\cos(\pi j y)\; dx\; dy & \text{for } j \neq 0, \\ c_{ij} & = & 4\int_0^1 \int_0^1 U(x, y, t_{start})\cos(\pi i x)\cos(\pi j y)\; dx\; dy & \text{for } i, j \neq 0. \end{array}$$

Immediately after sharpening,

$$U(x, y, t) = \begin{cases} 1 \text{ if } (x, y) \in R_t \\ 0 \text{ otherwise} \end{cases}$$

which implies that

$$c_{ij} = \gamma_{ij} \iint_{R_t} \cos(\pi i x)\cos(\pi j y)\; dA \tag{3.12}$$

where

$$\gamma_{ij} = \begin{cases} 1 & \text{if } i = j = 0 \\ 4 & \text{if } i \neq 0 \text{ and } j \neq 0 \\ 2 & \text{otherwise} \end{cases} \tag{3.13}$$

Thus, simple functions must be integrated over a complicated, non-rectangular region, R_t. This may be accomplished by recursively subdividing the domain (cf. [62, 61]), as we illustrate for the region, R, given in Figure 3.1a.

We begin by evaluating U at the corners of a number of equally-sized subregions, so as to capture the large-scale features of the shape. Typically, $n \times n$ subregions are selected because the corresponding U-values can be evaluated in just $\mathcal{O}(n^2 \log(n))$ operations using a fast Fourier transform (see, e.g., [17]). If the phase at all four corners of any subregion corresponds to white, then we assume that the subregion does not intersect with R and hence no contribution to the Fourier coefficients is made. This case corresponds to the subregions of Figure 3.1b which have at least one dashed edge. If all four corners of a subregion, $\tilde{Q}$, correspond to grey, however, we assume that $\tilde{Q} \subset R$ and add a contribution

$$\gamma_{ij} \iint\limits_{\tilde{Q}} \cos(\pi i x) \cos(\pi j y) \; dA$$

to each of the Fourier coefficients, c_{ij}, for $0 \leq i, j \leq n-1$. This case corresponds to the subregions of Figure 3.1b which have at least one thin, solid edge. Finally, if two phases occur, further subdivisions are carried out. We demonstrate this subdivision procedure for the subregion, Q, of Figure 3.1b.

Because Q is a mixed region, we divide it into quadrants, as shown in Figure 3.2b. Since the phase color at all corner points of quadrant Q_1^1 is white, we assume that this quadrant does not intersect with R and hence does not contribute to the Fourier coefficients. For each of the remaining quadrants, Q_1^2, Q_1^3 and Q_1^4, two phases occur, so

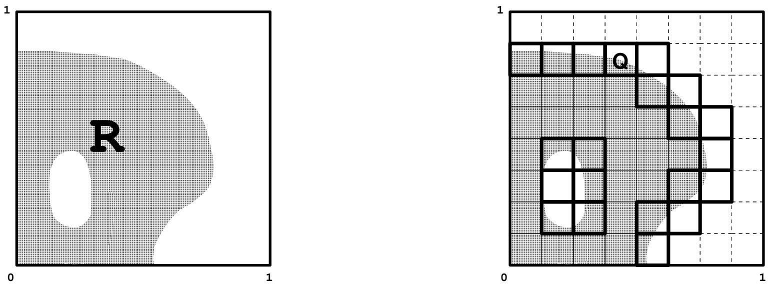


Fig. 3.1a. *Initial Region, R* Fig. 3.1b. *Coarsest Subdivisions*

Figure 3.1: Subdividing the Domain into its Coarsest Subregions

further subdivision is required. See Figure 3.2c.

Focusing on the refinement of the subregion, Q_1^3, we find that the phase of the upper right hand corner of Q_2^1 is different than that of the other corners. Thus, Q_2^1 is also subdivided. Corner points of the remaining subregions are grey, so we assume $Q_2^k \subset R$ for $k = 2, 3, 4$ and add contributions

$$\gamma_{ij} \iint\limits_{Q_2^k} \cos(\pi i x)\cos(\pi j y)\; dA$$

to each of the Fourier coefficients, c_{ij}, for $0 \leq i, j \leq n-1$.

Recursive subdivisions of the domain continue (see, e.g., Figure 3.2d) until regions containing multiple phases can be safely approximated by some simple numerical technique. The next section discusses methods for approximating the regions at the finest grid subdivisions.

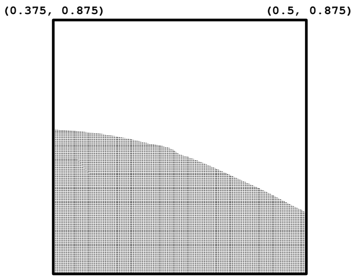


Fig. 3.2a. *Initial Subregion*

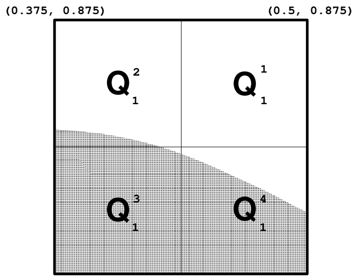


Fig. 3.2b. *One Subdivision*

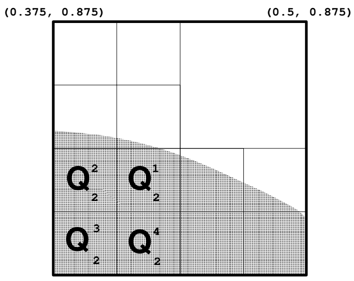


Fig. 3.2c. *Two Subdivisions*

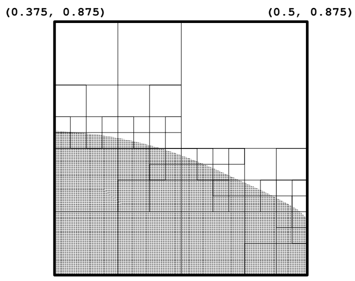


Fig. 3.2d. *Four Subdivisions*

Figure 3.2: Dividing a Subregion

3.3 Approximation of the Finest Subregions

In the previous section, a method was introduced for recursively dividing the domain into rectangles. At some point, however, we must stop subdividing and treat the finest cells. This section discusses how to approximate the contributions to the Fourier coefficients at the finest grid subdivisions.

3.3.1 Trivial Treatment of the Finest Subregions

An easy method for treating the finest grid subdivisions is to add one half the contribution that would occur for the whole subregion to each of the Fourier coefficients. In other words, a contribution

$$\frac{1}{2}\gamma_{ij} \int\int_{Q^k} \cos(\pi i x)\cos(\pi j y)\; dA$$

is added to $c_{ij}, 0 \leq i, j \leq n-1$, for each of the finest subregions, Q^k.

Often, this approach is unsatisfactory when accurate results in three dimensions are sought. If the finest subregions are of width h, then an $\mathcal{O}(h)$ error in the position of the front occurs. For smooth curves, we seek an $\mathcal{O}(\tau^2)$ approximation of the front (see next chapter), leading to $\mathcal{O}\left(\frac{1}{\tau^{2d-2}}\right)$ cells at the finest level of subdivision in d dimensions. Such a mesh is often impractical to treat, especially when three dimensional results are sought.

To achieve a more accurate approximation of regions and hence much faster results, we next consider piecewise linear approximations of the interface.

3.3.2 Piecewise Linear Approximation for Two-Phase Problems

In the last subsection, we saw that a trivial treatment of the integrals (3.12) at the finest grid subdivision produced an $\mathcal{O}(h)$ error in the position of the front, where h is the width of the finest grid subdivision. To produce an $\mathcal{O}\left(\frac{h^3}{\tau} + h^2\right)$ approximation of the interface, a simplicial decomposition of the region, R, with a piecewise linear approximation to the boundary can be used. We now describe such a method for two-phase problems in two and three dimensions.

Two Dimensional Problems

There are three main steps for approximating the integrals (3.12) over the finest grid subdivisions for two-phase problems in two dimensions. These are detailed below.

Step 1. Divide the Square Cell into Two Triangles.

We begin by breaking the square subdomain into two triangles and consider each separately (see, e.g., Figure 3.3). In two dimensions, this subdivision step is optional because it only slightly simplifies the implementation of Step 2.

Step 2. Approximate Regions Using Triangles.

We next approximate the desired phase with a number of triangular subregions. Details for this approximation method are now given for each of the four possible cases. (Five cases arise if Step 1 is omitted.)

Case 0. If none of the corners of the triangle belong to R, then we assume that R and the triangular subdomain do not overlap. No contribution to the Fourier coefficients is made.

Case 1. If one corner is in R, then linear interpolation is used to determine a triangular approximation to the subregion. For example, consider approximating the gray phase in $\triangle ABC$ of Figure 3.4. Letting $U(A), U(B)$ and $U(C)$ be the function values at points A, B and C, we approximate where the curve crosses the edges of the triangle by points a and b,

$$\begin{aligned} a &= B + \frac{\frac{1}{2} - U(B)}{U(C) - U(B)}(C - B), \\ b &= A + \frac{\frac{1}{2} - U(A)}{U(C) - U(A)}(C - A). \end{aligned} \tag{3.14}$$

This gives a triangular approximation, $\triangle baC$, to the desired subregion which can be used to approximate the contributions to the Fourier coefficients (see Step 3 below).

Case 2. If two corners are in R, then we linearly interpolate to estimate the location of the interface and break the subregion into two triangles. For example, the shaded region of $\triangle PQR$ in Figure 3.5 is approximated by $\triangle PpR$ and $\triangle Prp$, respectively.

Case 3. If three corners are in R, then we assume that the entire subdomain belongs to R, and we approximate the integrals (3.12) over the entire subdomain.

We seek an estimate of the error produced by this step for a smooth curve[1]. One source of error occurs when smooth curves are approximated by line segments. By Figure 3.6, this approximation produces an $\mathcal{O}(h^2)$ error in the position of the front, since the curvature is independent of h.

We also produce errors by replacing the actual front position with the interpolation (3.14). To determine this error, we consider the one dimensional analogue of curvature motion given by Figure 3.7. (We expect a similar result to hold in

[1] For affine velocity motions, nonsmooth corners may arise from singularities in the solution (see, e.g., [42]). Each corner can produce an $\mathcal{O}(h^2)$ error in the phase areas. However, because these corners are rapidly smoothed away, they typically do not affect the overall order of accuracy of the method.

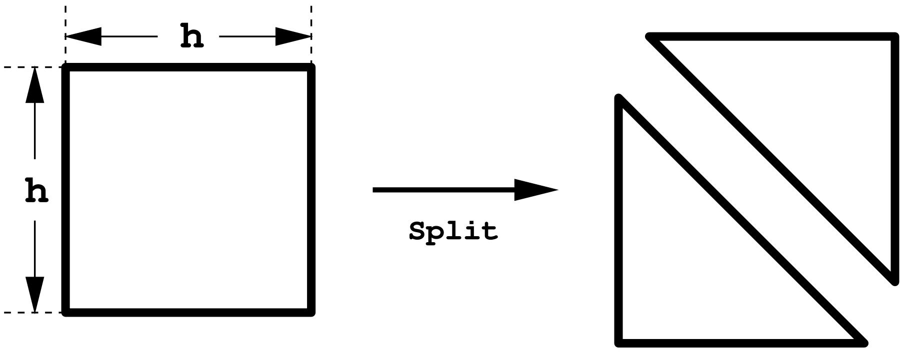


Figure 3.3: Subdividing a Square into Triangles

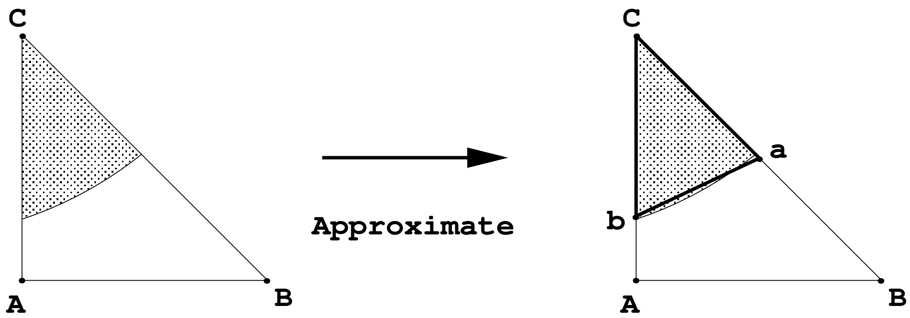


Figure 3.4: A Shape Represented by a Triangle

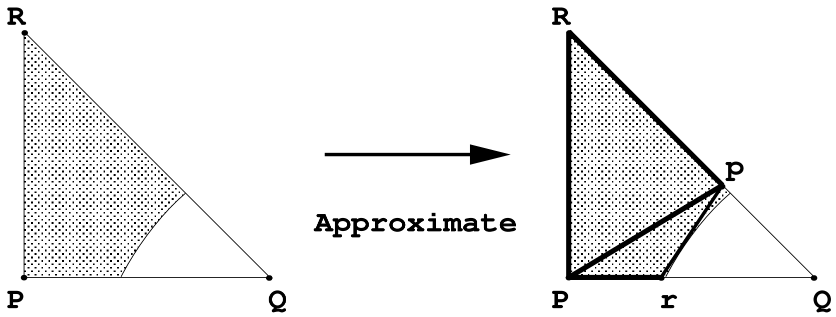


Figure 3.5: A Shape Represented by Triangles

two dimensions.) We seek an estimate of how well the interpolated value, $\tilde{x}_m$, approximates the front position, x_m; i.e., we seek an estimate of $|\tilde{x}_m - x_m|$.

Treating τ as a constant, an elementary Taylor series expansion gives us that

$$U(x_m) = U(x_1) + U'(x_1)(x_m - x_1) + \frac{1}{2}U''(\xi_1)(x_m - x_1)^2$$

where $\xi_1 \in (x_1, x_m)$. Replacing $U'(x_1)$ by a one-sided difference, we find

$$U(x_m) = U(x_1) + \frac{U(x_2) - U(x_1)}{(x_2 - x_1)}(x_m - x_1) - \frac{1}{2}U''(\xi_2)(x_2 - x_1)(x_m - x_1) + \frac{1}{2}U''(\xi_1)(x_m - x_1)^2$$

where $\xi_2 \in (x_1, x_2)$. Isolating x_m and subtracting the interpolated value, $\tilde{x}_m$, gives

$$\begin{aligned} |\tilde{x}_m - x_m| &= \left| \frac{-\frac{1}{2}U''(\xi_2)(x_2 - x_1)(x_m - x_1) + \frac{1}{2}U''(\xi_1)(x_m - x_1)^2}{U(x_2) - U(x_1)} \right| h, \\ &\leq \max_{\xi \in [x_1, x_2]} \left| \frac{U''(\xi)}{U(x_2) - U(x_1)} \right| h^3, \end{aligned} \tag{3.15}$$

where $h = x_2 - x_1$. Since $U(x) = \frac{1}{2} + \frac{1}{2}\mathrm{erf}\left(\frac{x - x_m}{2\sqrt{\tau}}\right)$ it is easy to show that

$$\begin{aligned} U(x_2) - U(x_1) &= \frac{1}{2\sqrt{\pi}} \exp\left(-\frac{1}{4t}(x_1 - x_m)^2\right) \left[\left(\frac{h}{\sqrt{\tau}}\right) + \mathcal{O}\left(\frac{h^3}{\tau^{3/2}}\right) \right], \\ U''(x) &= -\frac{1}{4\sqrt{\pi}} \exp\left(-\frac{1}{4t}(x - x_m)^2\right) \left(\frac{x - x_m}{\tau^{\frac{3}{2}}}\right). \end{aligned}$$

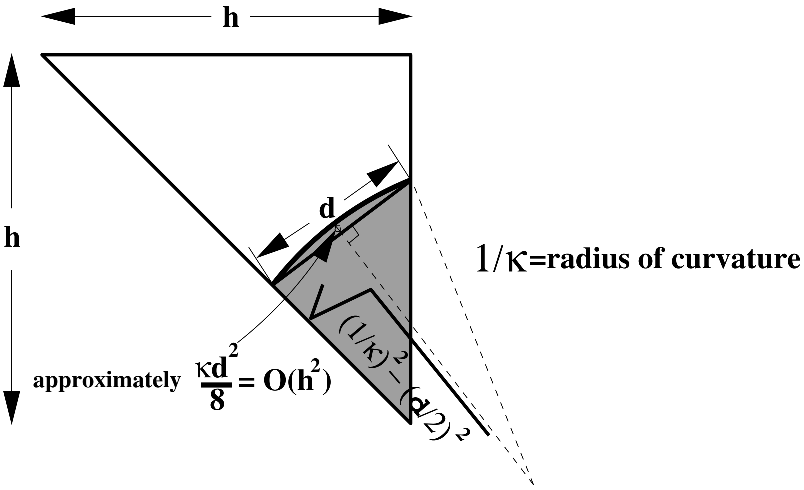


Figure 3.6: Errors Approximating Curved Segments

Applying these results to Equation (3.15) yields

$$|\tilde{x}_m - x_m| \leq \frac{1}{2}\left(\frac{h^2}{\tau}\right)\max_{\xi\in[x_1,x_2]}(\xi - x_m) + \text{h.o.t.}$$

Thus,

$$|\tilde{x}_m - x_m| = \mathcal{O}\left(\frac{h^3}{\tau}\right).$$

Taking into account both of the contributions to the error, we find that this triangular approximation of regions produces an $\mathcal{O}\left(\frac{h^3}{\tau} + h^2\right)$ error in the position of the front.

Step 3. Integrate over each Triangular Subregion.

We are now left with the task of adding a contribution

$$I_{ijk} = \gamma_{ij} \int\int_{T_k} \cos(\pi i x)\cos(\pi j y)\, dA \tag{3.16}$$

to each Fourier coefficient, c_{ij}, for each triangular subregion, T_k.

Expanding the integrand about some point, $(\tilde{x}, \tilde{y})$, in T_k yields

$$I_{ijk} = \gamma_{ij}\text{Area}(T_k)\cos(\pi i \tilde{x})\cos(\pi j \tilde{y})$$

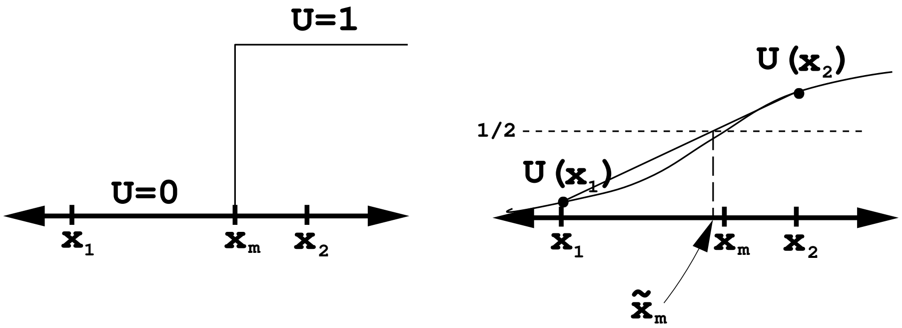


Fig. 3.7a. *Sharpened Intervals* Fig. 3.7b. *After a Time, τ*

Figure 3.7: The Interpolation Step

$$
\begin{aligned}
- \quad & \pi \iint_{T_k} [j \cos(\pi i \tilde{x}) \sin(\pi j \tilde{y})(y - \tilde{y}) + i \sin(\pi i \tilde{x}) \cos(\pi j \tilde{y})(x - \tilde{x})] \; dA \\
+ \quad & \mathcal{O}([i^2 + j^2] h^4)
\end{aligned}
$$

where Area(T_k) is the area of triangle T_k. Because the integral term cancels when we expand about the centroid of the triangle[2], we choose to approximate each contribution (3.16) by

$$I_{ijk} \approx \gamma_{ij} \mathrm{Area}(T_k) \cos(\pi i \tilde{x}) \cos(\pi j \tilde{y}) \tag{3.17}$$

where $(\tilde{x}, \tilde{y})$ is the centroid of T_k. This approximation is preferred over the direct evaluation of the integrals (3.16) because it is much faster (it only requires two trigonometric evaluations) and it produces errors which are typically small relative to those arising in Step 2.

[2]The centroid of a triangle with corners A, B and C is $\frac{A+B+C}{3}$.

Three Dimensional Problems

There are also three steps for approximating the contributions to the Fourier coefficients over the finest grid subdivisions for two-phase problems in three dimensions. These are outlined below.

Step 1. Divide the Cube into Six Tetrahedrons.

We begin by breaking cube-shaped subdomains into six tetrahedrons and consider each separately. For example, the cube in Figure 3.8 would be split into the tetrahedrons, $T_{ABFD}, T_{AEFD}, T_{DHEF}, T_{CBFD}, T_{CGFD}$ and T_{DHGF}. In three dimensions, this subdivision step is highly recommended because it significantly simplifies the implementation of Step 2.

Step 2. Approximate Regions Using Tetrahedrons.

We next approximate the desired phase with a number of tetrahedrons. For smooth surfaces[3], this step produces an $\mathcal{O}\left(\frac{h^3}{\tau} + h^2\right)$ error in the position of the front where h is the width of the finest grid subdivision. An outline of this approximation method is now given for each of the five possible cases. (Nine cases arise if Step 1 is omitted.)

Case 0. If none of the corners of the tetrahedron belong to R, no contribution to the Fourier coefficients is made.

Case 1. If one corner is in R, then the region is approximated by a tetrahedron. For example, the shaded region in tetrahedron T_{ABCD} of Figure 3.9 would be approximated by T_{abcD}.

[3]Nonsmooth corners may arise from singularities in the solution (see, e.g., [35]). Each corner can produce an $\mathcal{O}(h^3)$ error in the phase volumes. However, because these corners are rapidly smoothed away, they typically do not affect the overall order of accuracy of the method.

Case 2. If two corners are in R, then we linearly interpolate to estimate the location of the interface and break the subregion into three tetrahedrons. For example, the shaded region of T_{ABCD} in Figure 3.10 is approximated by the tetrahedrons T_{Begf}, T_{Bfgh} and T_{BhDg}, respectively.

Case 3. If three corners are in R, then we represent the shape as the difference of shapes which are treated using Cases 1 and 4. See Figure 3.11 for an example.

Case 4. If four corners are in R, then we assume that the entire subdomain belongs to R, and we approximate the integrals (3.18) over the entire subdomain.

Step 3. Integrate over each Tetrahedron.

For each tetrahedron, T_ℓ, a contribution

$$I_{ijk\ell} = \gamma_{ijk} \iiint\limits_{T_\ell} \cos(\pi i x)\cos(\pi j y)\cos(\pi k z)\; dV \tag{3.18}$$

where

$$\gamma_{ijk} = \begin{cases} 1 & \text{if } i = j = k = 0 \\ 2 & \text{if exactly two of } i, j \text{ or } k \text{ equals } 0 \\ 4 & \text{if exactly one of } i, j \text{ or } k \text{ equals } 0 \\ 8 & \text{otherwise} \end{cases}$$

must be added to each Fourier coefficient, c_{ijk}.

Similar to the two dimensional case, we expand the integrand about the centroid of the tetrahedron[4], to arrive at

$$I_{ijk\ell} \approx \gamma_{ijk}\text{Volume}(T_\ell)\cos(\pi i\tilde{x})\cos(\pi j\tilde{y})\cos(\pi k\tilde{z})$$

where $(\tilde{x}, \tilde{y}, \tilde{z})$ is the centroid of T_ℓ and Volume(T_ℓ) is its volume. This approximation is preferred over the direct evaluation of the integrals (3.18) because it is

[4]The centroid of a tetrahedron with corners A, B, C and D is $\frac{A+B+C+D}{4}$.

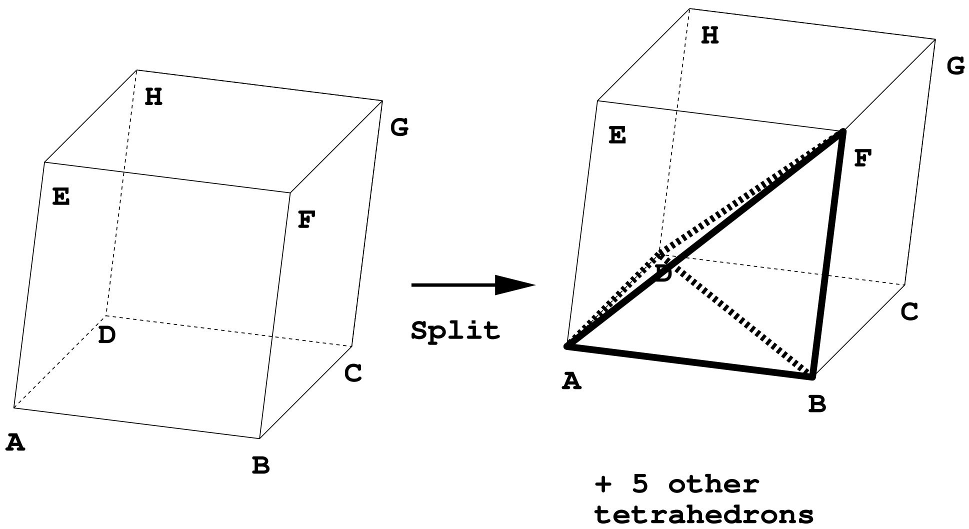


Figure 3.8: Subdividing a Cube into Tetrahedrons

much faster (it only requires three trigonometric evaluations) and it produces errors which are typically small relative to those arising in Step 2.

3.3.3 Piecewise Linear Approximations for Junctions

The previous subsections do not explain how to approximate contributions to the Fourier coefficients when junctions occur. Several methods for carrying out such approximations are possible. This subsection describes three of these methods for two-dimensional problems; extensions to three dimensions are straightforward.

Method 1. Trivial Treatment of the Junction.

If all corners of the triangular subregion correspond to a different phase (e.g., $\triangle ABC$ of Figure 3.12 or $\triangle ACD$ or Figure 3.13) then $\frac{1}{3}$ the contribution (3.12) that would occur for the entire subregion is added to each of the three sets of Fourier coefficients. Other regions are assumed to contain only two phases and hence are treated according to the discussion of the previous subsection.

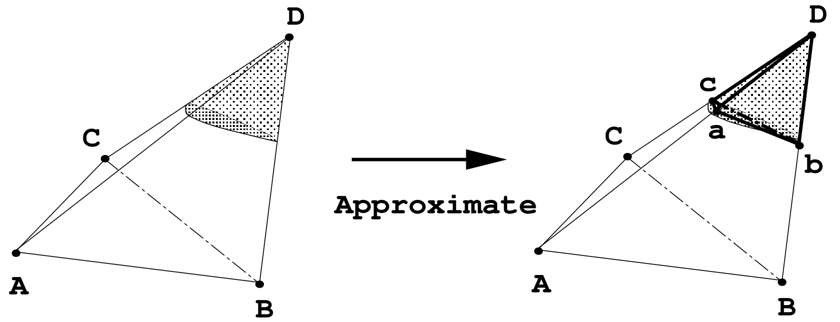


Figure 3.9: A Shape Approximated by a Tetrahedron

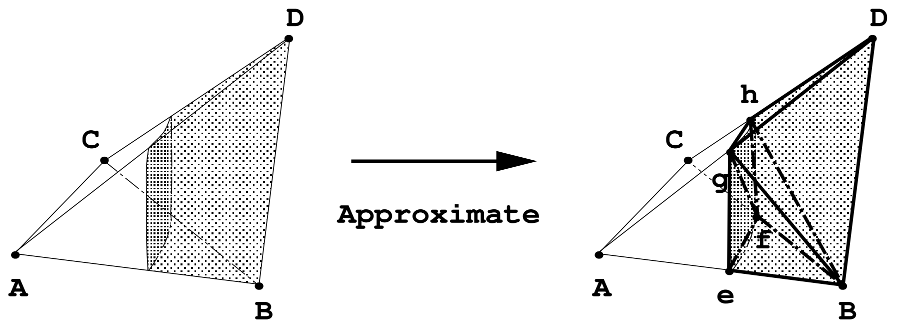


Figure 3.10: A Shape Approximated by Tetrahedrons

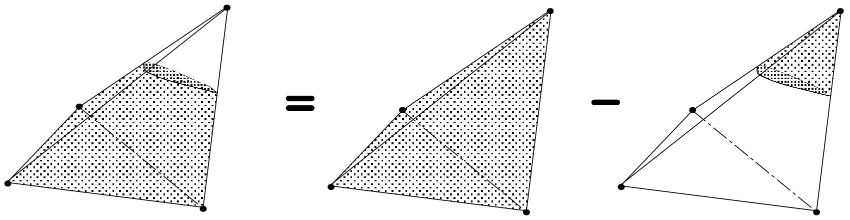


Figure 3.11: A Shape Represented as a Difference

Although the method is somewhat crude, it only produces an $\mathcal{O}(h^2)$ error in the phase areas for cells which possess three phases. Thus, the overall order of accuracy of the method is not degraded since the number of junctions present in two dimensions is independent of h.

Method 2. Triangular Approximation of Regions.

We may also use several triangles to approximate regions that contain junctions. If different phases occur at each corner of a triangular region, then that region can be broken into three triangles which have at most two phases each. These triangular subregions are treated according to the discussion of the previous subsection.

For example, $\triangle ABC$ or Figure 3.12 can be broken into $\triangle baC$, $\triangle Aab$ and $\triangle ABa$ where a and b are found by linear interpolation,

$$\begin{aligned} a &= \frac{U_2(B)-U_1(B)}{U_2(B)-U_2(C)+U_1(C)-U_1(B)}C + \frac{U_1(C)-U_2(C)}{U_2(B)-U_2(C)+U_1(C)-U_1(B)}B, \\ b &= \frac{U_2(A)-U_3(A)}{U_2(A)-U_2(C)+U_3(C)-U_3(A)}C + \frac{U_3(C)-U_2(C)}{U_2(A)-U_2(C)+U_3(C)-U_3(A)}A, \end{aligned}$$

and $U_i(X), 1 \leq i \leq 3$, is the value of U_i at the point X.

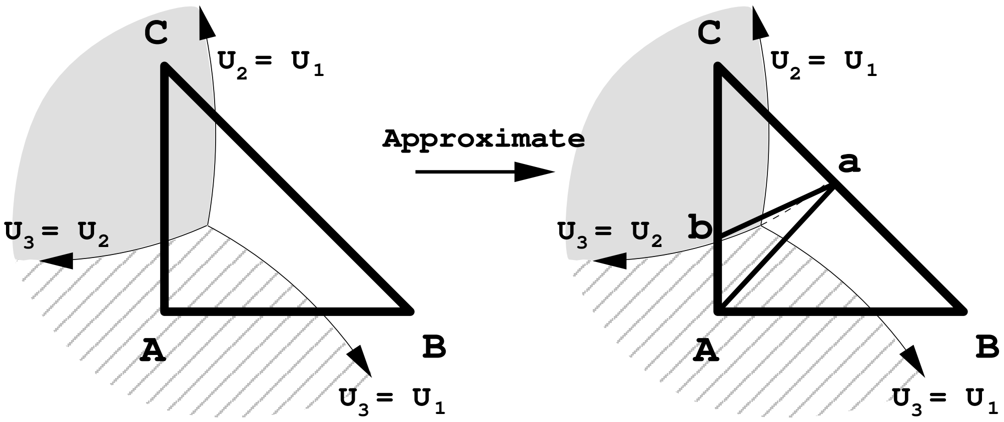


Figure 3.12: Junction for which all Corner Phases Differ

The other possible junction occurs when exactly two different phases are present at the corners of the subdomain (e.g., $\triangle ABC$ of Figure 3.13). In this case, we break the region into three triangles, as we would for the two-phase case (see, e.g., Figure 3.13). The triangles that arise contain two phases, and hence should be treated according to the discussion of the previous subsection.

Method 3. Subdividing Regions Containing Multiple Phases.

The final approach that we mention is simply to subdivide any region containing more than two phases. After a few iterations, the smallest subregions that arise can be treated by Method 1 to give an accurate approximation of each integral.

Our implementations have used the first method (i.e., the trivial treatment), since this straightforward approach typically produces errors which are very similar to those for the other, more involved methods. For example, in the three-phase problem displayed in Figure 4.6, cells containing multiple phases contribute less than 2% of the total error arising from the spatial discretization.

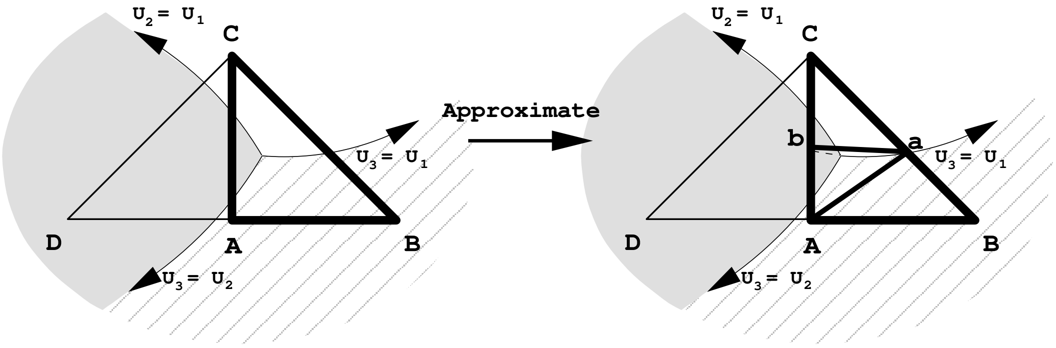


Figure 3.13: Two Phases Represented at Corners

3.4 Refinement Techniques

In Section 3.2, a recursive algorithm for subdividing the domain was introduced. We now carry out a more detailed study of the method and introduce a gradual refinement which overcomes certain limitations of the original algorithm.

For illustrative purposes, all examples set the width of the coarsest grid to be $H = \frac{1}{8}$. Similar results arise for the usual choice of $H = \frac{1}{n}$.

3.4.1 The Original Refinement Algorithm

The original refinement algorithm of Section 3.2 is effective for a variety of problems. Application of this method to the two-phase shape of Figure 3.14a, for example, produces a refinement which captures the entire interface at the level of the finest grid subdivision. See Figure 3.14b.

For certain smooth regions, however, small slivers can be missed when the algorithm is applied. Consider, for example, a translation by $\left(\frac{19}{1024}, \frac{5}{64}\right)$ of the shape found in Figure 3.14a. Applying the subdivision algorithm to the translated shape gives the mesh displayed in Figure 3.15a. A magnification of the leftmost part of the shape (see Figure 3.15b) indicates that a small, thin region is missed by the algorithm.

We would like at least a crude estimate of the error arising from these slivers. Suppose that the local curvature of the boundary for such a region is κ. By Figure 3.16, this leads to an $\mathcal{O}(\kappa H^3)$ error in the corresponding phase area. The number of coarse-grid cells that neglect these slivers will be problem-dependent. For a convex shape, however, only the cells at the extreme top, bottom, left and rightmost parts of the curve need to be considered. As the shape shrinks under mean curvature motion, we expect that the proportion of time that slivers of width $\mathcal{O}(H^2)$ arise in any of these four cells will be $\mathcal{O}(H)$. Assuming $H = \frac{1}{n}$ and $n = \mathcal{O}(\frac{1}{\sqrt{\tau}})$ (see inequality (3.11)), the total neglected area over $\mathcal{O}(\frac{1}{\tau})$ steps of the method will be $\mathcal{O}(\tau)$, on average.

The original refinement algorithm also produces errors when applied to nonsmooth shapes. Consider, for example, the region displayed in Figure 3.17. Such a shape may arise for an affine velocity motion when a topological breaking occurs. (Note that such topological changes do not occur for mean curvature motion in two dimensions [31, 36].) Applying the original subdivision algorithm to the shape gives the mesh displayed in Figure 3.18a. Clearly, an $\mathcal{O}(H^2)$ error in the phase area is produced at the cell containing the sharp corners. This corresponds to an $\mathcal{O}(\tau)$ error when $H = \frac{1}{n}$ and n is chosen according to (3.11).

For the numerical experiments that we have considered, this flaw in the refinement technique produces errors which are no larger than the $\mathcal{O}(\tau)$ errors which arise from the MBO-method for smooth, two phase problems and are smaller than the $\mathcal{O}(\sqrt{\tau})$ which arise for junctions (see next chapter). Nonetheless, we introduce a more accurate refinement in the next subsection to give us a greater confidence in our results, especially for complicated, non-convex initial regions. A more accurate refinement is also of value when higher order, extrapolated methods are considered (see next chapter).

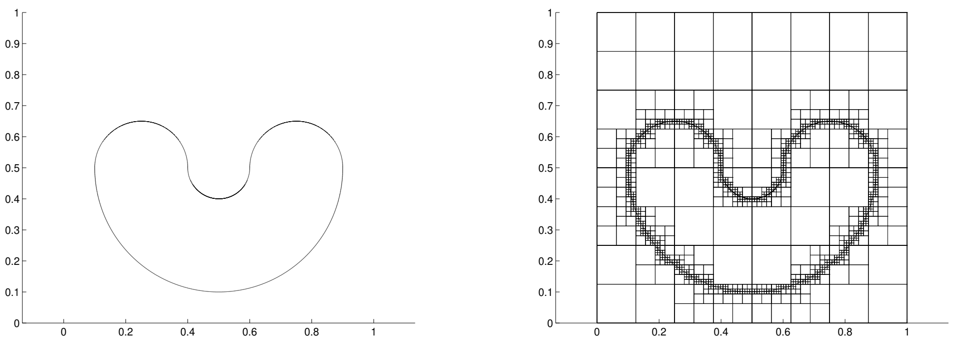


Fig. 3.14a. *A Smooth Shape* Fig. 3.14b. *Original Refinement*

Figure 3.14: Refinement of a Smooth Region

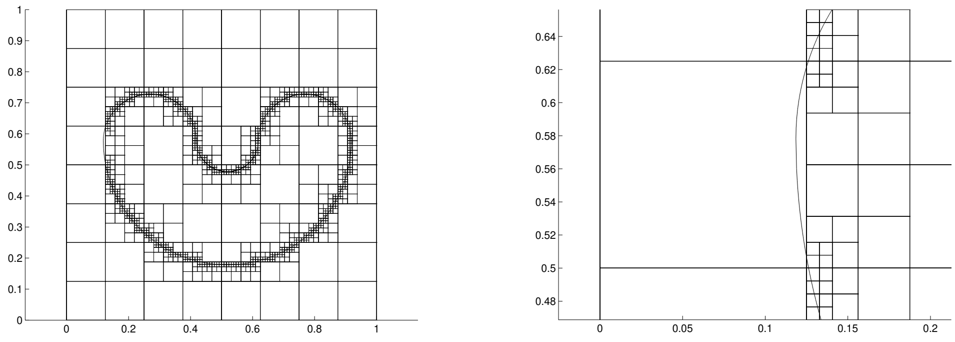


Fig. 3.15a. *Refined Domain* Fig. 3.15b. *Zoom-in Near a Neglected Section*

Figure 3.15: Original Refinement can miss Slivers of the Region

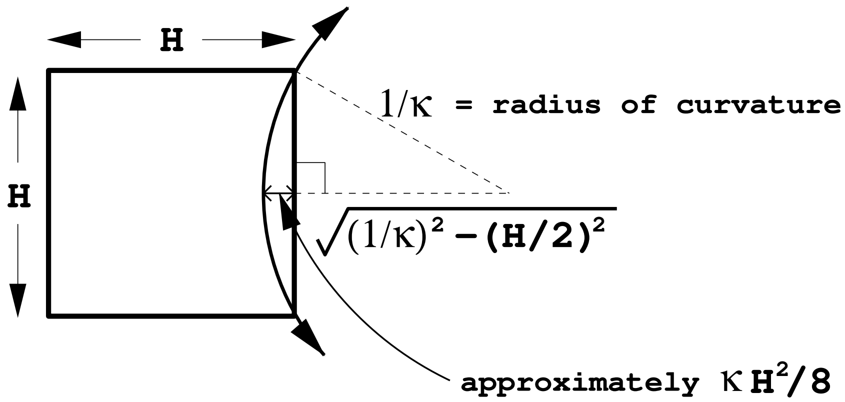


Figure 3.16: A Neglected Section

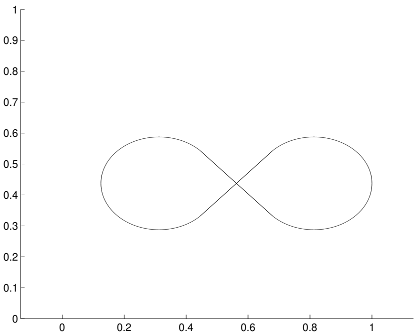


Figure 3.17: A Problem with Sharp Corners

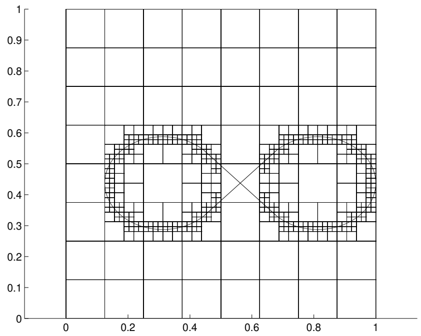


Fig. 3.18a. *Original Refinement*

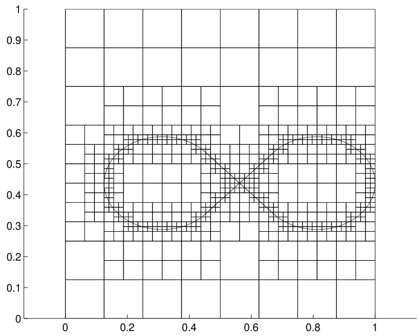


Fig. 3.18b. *Gradual Refinement*

Figure 3.18: Refinement Methods for Corners

3.4.2 A Method for a Gradual Refinement

As we have seen in the previous subsection, the original subdivision algorithm can miss small pieces of both smooth and nonsmooth shapes. We seek a refinement which captures the entire interface at the level of the finest grid subdivision, even for nonsmooth shapes.

To achieve this objective, a gradual refinement was implemented. This method proceeds according to the original subdivision algorithm of Section 3.2, with the following additional consideration:

> Whenever any cell is refined, check the subdivision level of the neighboring cells. Subdivide neighbors which are two or more levels of refinement coarser.

This method accurately represents the narrow, sliver-shaped regions that were missed using the original refinement. By using a fine subdivision in a small neighborhood of the interface, this method even captures the rapid variations in the front that arise from corners. See Figures 3.19 and 3.18b for examples.

Certainly, this gradual refinement produces more cells than the original approach. The order of the number of cells is unchanged, however. To see this, note that cells of width

$$h_\ell = 2^{-\ell} H, \text{ where } 0 < \ell \leq \log_2 \left(\frac{H}{h}\right)$$

form a band at most two cells wide on each side of the interface. The length of each band can be bounded by a constant, K, independent of h (e.g., bands for a convex region are shorter than the perimeter of the domain). Letting $n_{\tilde{h}}$ be the number of cells of width $\tilde{h}$, we observe that

$$\begin{aligned}
\text{Total number of cells} &= n_h + n_{2h} + \ldots + n_{\frac{H}{2}} + n_H, \\
&< \frac{4K}{h} + \frac{4K}{2h} + \ldots + \frac{4K}{\frac{H}{2}} + n^2, \\
&< \frac{8K}{h} + n^2.
\end{aligned}$$

Thus, $\mathcal{O}(\frac{1}{h} + n^2)$ cells are required, which matches in order the result for the original refinement.

Implementation of this gradual refinement is somewhat more involved than the original approach because cell neighbors must be found. Many data structures appropriate for this task have been considered [62, 61]. Our implementation defines the grid as a list of vertices (cf. [28]), each of which is described by a data structure:

structure vertex

```
BEGIN
x, y;          % Coordinates of the vertex.
u;             % Function value at this vertex.
n, s, e, w;    % Pointers to the vertices north (above), south (below), east (right)
                 and west (left) of this vertex.
h;             % If there is no cell northeast of this vertex, set to (-1).
                 Otherwise, set to the width of the cell northeast of this vertex.
END
```

Access of cells and their neighbors is carried out indirectly by traversing their vertices. For example, the refinement of region Q in Figure 3.20a can be carried out as follows:

1. Determine the cell width, h, from the data structure defining vertex, v_4. Starting at v_4, traverse the perimeter of Q by going north, east, south and finally west a distance h.

2. From the traversal in Step 1, it is clear that no vertex should be added to edge (v_4, v_7). Thus, four vertices, v_9, v_{10}, v_{11} and v_{12}, are added to the grid as shown in Figure 3.20b. Updates to the data structures for all boldfaced vertices in Figure 3.20b must also be carried out.

3. Finally, we check if neighboring cells need to be refined. Since there are vertices immediately north of v_7 and v_8, the cell north of Q is at most one level coarser than the refined regions. Thus, refinement north of Q is unnecessary. Similarly, no refinement occurs east or west of Q. There is no vertex immediately south of v_5, however, so a refinement of the region south of Q is needed. This is carried out by applying steps 1 to 3 to the region northeast of vertex v_1.

3.5 Fast, Transform-Based Algorithms

The refinements of Sections 3.2 and 3.4 lead to a large number of function evaluations,

$$U(x, y) = \sum_{j,j'=0}^{n-1} c_{jj'} \exp(-\pi^2[j^2 + (j')^2]\tau) \cos(\pi j x) \cos(\pi j' y). \tag{3.19}$$

Because these evaluations occur on an unequally spaced grid, a fast Fourier transform cannot be used. Direct evaluation of Equation (3.19) at N_q points, however, is often prohibitively expensive because $\mathcal{O}(n^2 N_q)$ operations are required. Similarly, evaluation

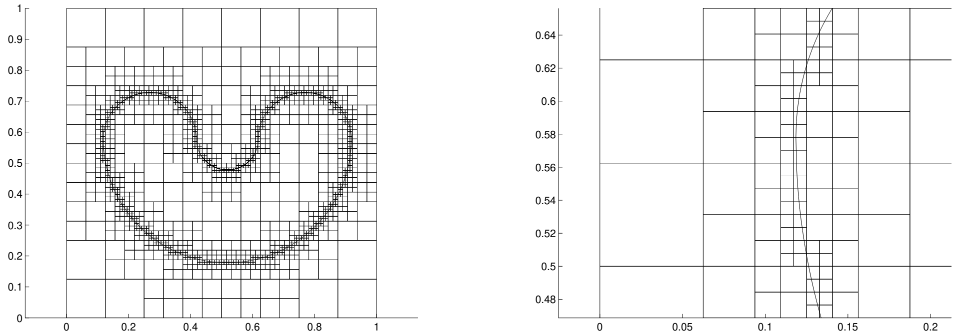


Fig. 3.19a. *Refined Domain* Fig. 3.19b. *Zoom-in near a Difficult Section*

Figure 3.19: Gradual Refinement Captures the Entire Region

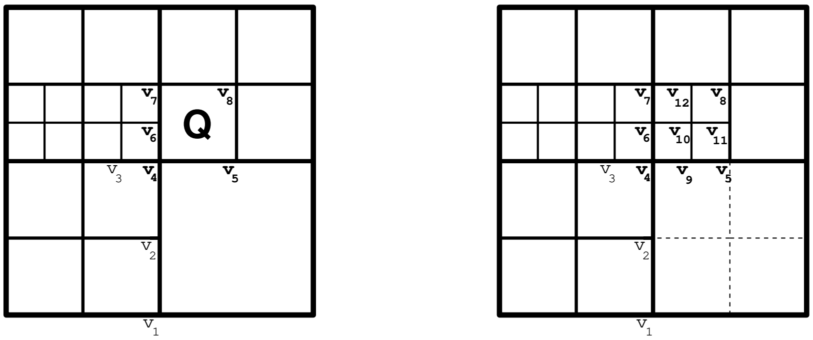


Fig. 3.20a. *Initial Domain* Fig. 3.20b. *After Subdividing Q*

Figure 3.20: Refining a Cell

of the Fourier coefficients using Equations (3.12) and (3.17) leads to a Fourier sum of the form

$$c_{jj'} = \sum_{\ell=0}^{N_p-1} d_\ell \cos(\pi j x_\ell)\cos(\pi j' y_\ell) \tag{3.20}$$

where $0 \leq j, j' \leq n-1$ and (x_ℓ, y_ℓ) are unequally spaced. Once again, a fast Fourier transform cannot be used, and direct evaluation leads to $\mathcal{O}(n^2 N_p)$ operations.

In this section, we consider recent methods for the fast evaluation of (3.19) and (3.20). Specifically, we discuss an unequally spaced fast Fourier transform method [5] that we have applied to the new, spectral method. This transform leads to an algorithm that typically requires only

$$\mathcal{O}\left(\frac{1}{\tau}\log^2(\tau)\right)$$

operations per τ-step for the basic MBO-method.

For consistency with [5], our discussion will assume that Equations (3.19) and (3.20) have been re-arranged as exponential sums. Specifically, we express Equation (3.19) as

$$U(x,y) = \sum_{j,j'=-n+1}^{n-1} f_{jj'} \exp(-i\pi j x)\exp(-i\pi j' y) \tag{3.21}$$

where

$$f_{jj'} = \frac{1}{\gamma_{jj'}} c_{|j||j'|} \exp(-\pi^2[j^2 + (j')^2]\tau)$$

and $\gamma_{jj'}$ is given by Equation (3.13). Equation (3.20) is re-written as

$$c_{jj'} = \frac{1}{2}\mathrm{Real}(S_{jj'}) + (-1)^j \frac{1}{2}\mathrm{Real}(\tilde{S}_{jj'}) \tag{3.22}$$

where

$$S_{jj'} = \sum_{\ell=0}^{N_p-1} d_\ell \exp(-i\pi j x_\ell)\exp(-i\pi j' y_\ell), \tag{3.23}$$

$$\tilde{S}_{jj'} = \sum_{\ell=0}^{N_p-1} d_\ell \exp(-i\pi j[1-x_\ell])\exp(-i\pi j' y_\ell), \tag{3.24}$$

and $0 \leq j, j' \leq n-1$. (Note that it is easy to show that Equation (3.22) is symmetrical with respect to x and y by factoring $e^{-i\pi j} = (-1)^j$ from Equation (3.24).)

3.5.1 Overview

Several methods for the fast evaluation of Equations (3.21) and (3.22) have been developed. Lagrange interpolation has been used to replace function values at arbitrary points by several function values on an equally spaced grid [55, 10]. Taylor expansions have also been used to correct for deviations from an equally spaced grid [70]. Although these methods produce a significant speed-up over a direct approximation of the summations, they are inherently inefficient because they must oversample or compute on a much finer than $n \times n$ mesh to accurately evaluate all n^2 Fourier modes. Furthermore, no studies have been undertaken to determine how the accuracy and speed of these methods depends on the oversampling factor and the choice of interpolation [5].

Recent methods do not need a significant amount of oversampling and have been proven to converge quickly. In [5], for example, a method based on multiresolution analysis was developed and implemented that evaluates Equation (3.21) at N_q points in

$$\mathcal{O}\left(N_q \log^2\left(\frac{1}{\epsilon}\right) + n^2 \log(n)\right) \tag{3.25}$$

operations, and evaluates all n^2 Fourier coefficients of (3.22) in

$$\mathcal{O}\left(N_p \log^2\left(\frac{1}{\epsilon}\right) + n^2 \log(n)\right) \tag{3.26}$$

operations, where ϵ is the precision of the computation. Essentially the same bounds are obtained in [24] for an interpolation using Gaussian Bells and in [68] for an algorithm which uses Lagrange interpolation and Green's theorem.

In practice, numerical experiments [24, 68, 5] suggest that Beylkin's algorithm [5] is the fastest of the three recent methods. For this reason, we next consider Beylkin's approach for evaluating Equations (3.21) and (3.22).

3.5.2 The Unequally Spaced Fast Fourier Transform

In this subsection, we outline Beylkin's algorithm [5] for evaluating Equations (3.21) and (3.22). We have implemented this algorithm in two dimensions; extensions to three dimensions are possible [5].

The version of the algorithm given in [5] requires that grid points must not occur within a distance $\mathcal{O}\left(\frac{1}{n}\right)$ of the boundary of the domain $[0,1] \times [0,1]$. In the following implementation, we have scaled and translated points to overcome this limitation. A more elegant and faster version can be derived using periodic extensions of the Fourier coefficients given in Equations (3.21) and (3.22) [5].

Fast Evaluation of Fourier Coefficients

Three steps are required for the evaluation of Fourier coefficients using Beylkin's algorithm. These are described below for Equation (3.23). Equation (3.24) can be treated by replacing x_ℓ in step 1 by $(1 - x_\ell)$.

1. The algorithm begins by projecting an integral representation of the sum onto the span of a number of translated central B-splines. Specifically, we evaluate

$$f_{kk'} = \sum_{\ell=0}^{N_p-1} d_\ell e^{-\frac{1}{2}\pi i n(x_\ell + y_\ell)} \beta^{(m)}\left(nx_\ell + \frac{1}{2}n - k\right) \beta^{(m)}\left(ny_\ell + \frac{1}{2}n - k'\right)$$

for $0 \leq k, k' \leq 2n-1$. The m^{th}-order central B-spline, $\beta^{(m)}$, can be stably evaluated using the recursion

$$\beta^{(m)}(x) = \frac{\frac{1}{2}(m+1)+x}{m}\beta^{(m-1)}\left(x+\frac{1}{2}\right) + \frac{\frac{1}{2}(m+1)-x}{m}\beta^{(m-1)}\left(x-\frac{1}{2}\right)$$

where

$$\beta^{(0)}(x) = \begin{cases} 1 & \text{if } x \in (\frac{1}{2}, \frac{1}{2}] \\ 0 & \text{otherwise.} \end{cases}$$

See [7] for an algorithm that evaluates each of the components of f in only $\mathcal{O}(m^2)$ operations.

2. Using a fast Fourier transform, we next evaluate

$$F_{jj'} = \sum_{k=0}^{2n-1} \sum_{k'=0}^{2n-1} f_{kk'} e^{-2\pi ikj/(2n)} e^{-2\pi ik'j'/(2n)}$$

for $-n \leq j, j' \leq n$.

3. A final scaling gives an approximation to the desired coefficients

$$S_{j+\frac{n}{2},j'+\frac{n}{2}} \approx \frac{e^{\frac{\pi i}{2}(j+j'+n)}}{\sqrt{a^{(m)}(j/(2n))a^{(m)}(j'/(2n))}} F_{jj'}$$

where

$$a^{(m)}(\xi) = \sum_{\ell=-m}^{\ell=m} \beta^{(2m+1)}(\ell) e^{2\pi i\ell\xi},$$

and $-\frac{n}{2} \leq j, j' \leq \frac{n}{2} - 1$.

Summing over steps 1 to 3, we see that a total of

$$\mathcal{O}(m^2 N_p + n^2 \log(n))$$

operations are taken. A proof of the operation count (3.26) can be derived using the relationship between m and the precision of the calculation, ϵ [5]. Our implementations set $m = 23$, since this choice has been experimentally shown to give errors which are similar to those arising from roundoff in double precision calculations (see [5]).

Fast Evaluation of Fourier Summations

Beylkin also gives a three step algorithm for the fast evaluation of Fourier sums [5]. This method is described below for the evaluation of Equation (3.21).

1. The algorithm begins by modifying the Fourier coefficients according to
$$\hat{F}_{kk'} = \frac{w_{kk'}}{\hat{b}_k \hat{b}'_k}$$
where
$$\begin{aligned} w_{kk'} &= \begin{cases} f_{kk'} & \text{for } -n+1 \le k, k' \le n-1, \\ 0 & \text{otherwise} \end{cases} \\ \hat{b}_k &= \sum_{\ell=-(m-1)/2}^{(m-1)/2} \beta^{(m)}(\ell) e^{2\pi i \ell k/(4n)} \end{aligned}$$
and $-2n \le k, k' \le 2n-1$.

2. Using a fast Fourier transform and the result from step 1,
$$F_{jj'} = \sum_{k=-2n}^{2n-1} \sum_{k'=-2n}^{2n-1} \hat{F}_{kk'} e^{-2\pi i k j/(4n)} e^{-2\pi i k' j'/(4n)}$$
is evaluated for $-2n \le j, j' \le 2n-1$.

3. Having completed the pre-processing steps 1 and 2, an arbitrary number of function evaluations may be carried out using
$$U(x,y) = \sum_{j,j'=-2n}^{2n-1} F_{jj'} \beta^{(m)}(2nx - j) \beta^{(m)}(2ny - j'). \tag{3.27}$$
Using the algorithm discussed in [7], each of these evaluations requires only $\mathcal{O}(m^2)$ operations.

Summing over all three steps, we find that
$$\mathcal{O}(m^2 N_q + n^2 \log(n))$$
operations are taken to produce N_q function evaluations. This corresponds to the work estimate given in (3.25).

Application to the New, Spectral Method

We seek an estimate of how much work arises from an iteration of the new, spectral method when these transform methods are applied:

1. We begin by carrying out the first two steps of Beylkin's method for the fast evaluation of Fourier sums. This step produces a total of $\mathcal{O}(n^2 \log n)$ operations.

2. The domain is refined according to Sections 3.2 and 3.4. Since each evaluation of the function, U, can be carried out in $\mathcal{O}(m^2)$ operations using Equation (3.27), refinement produces a total operation count of (3.25).

3. We next collect the contributions to each Fourier coefficient into a sum (3.20) in $\mathcal{O}(N_p)$ operations. The number of operations to evaluate this sum according to Beylkin's method is given by (3.26).

Summing over all three steps, we find that

$$\mathcal{O}\left(N_p \log^2(\epsilon) + N_q \log^2(\epsilon) + n^2 \log(n)\right)$$

operations are taken where ϵ is the precision of the calculation. Using the fact that $\mathcal{O}\left(\frac{1}{h}\right)$ refined cells arise (see Section 3.4.2), it is clear that $N_q = \mathcal{O}\left(\frac{1}{h}\right)$ and $N_p = \mathcal{O}\left(\frac{1}{h}\right)$. The remaining $\mathcal{O}(n^2)$ coarse grid cells may be treated with a fast Fourier transform in $\mathcal{O}(n^2 \log(n))$ operations. Applying these relationships, along with $h \ll \frac{1}{n}$, we see that a total of

$$\mathcal{O}\left(\left(\frac{1}{h}\right) \log^2(h) + n^2 \log(n)\right)$$

operations arise at each iteration of the spectral discretization of the MBO-method. As we shall see in the next chapter, the basic MBO-method produces an $\mathcal{O}(\tau^2)$ error in the position of a smooth curve at each step of the method. To avoid degrading this accuracy (see Section 3.3.2), we select $h = \mathcal{O}(\tau)$ to arrive at

$$\mathcal{O}\left(\frac{1}{\tau}\log^2(\tau)\right)$$

operations per step. For the case of junctions, we may apply the same considerations to determine that

$$\mathcal{O}\left(\frac{1}{\tau}\log(\tau)\right)$$

operations are required per step to avoid degrading the overall accuracy of the method.

3.6 Comparison to the Usual Finite Difference Discretization

There are several reasons why the spectral method described in this chapter is preferred over the usual finite difference approach. These reasons are outlined below.

1. As has been discussed in Section 3.1, only low frequency modes need to be approximated provided τ is not taken very small. A large amount of computational work is saved by only treating these low frequency modes.

2. The new, spectral method does not require any time-stepping between t_{start} and $t_{start} + \tau$. This eliminates a possible source of error and produces large savings in computational work.

3. Local refinement is much simpler to implement for the new, spectral approach because it is done in the context of a quadrature, rather than a discretization of a differential equation.

4. By using a spectral method, the error arising from discretizing the heat equation can be nearly eliminated. This is an attractive feature, because it makes extrapolation in τ practical (see next chapter), which in turn allows for larger τ-steps. When

larger τ-steps are taken, even fewer basis functions are required to solve the heat equation to a given accuracy.

5. The original finite difference algorithm must satisfy (2.7) globally, or part of the front may erroneously remain stationary. By recursively refining near the interface, the new, spectral approach can essentially eliminate this restriction.

6. The new, spectral method also gives an $\mathcal{O}\left(\frac{h^3}{\tau} + h^2\right)$ approximation of the location of the front, which is greatly superior to the first order approximation arising for finite differences. As we saw in the previous section, this improved accuracy, in part, explains why
$$\mathcal{O}\left(\frac{1}{\tau}\log^2(\tau)\right)$$
operations are needed per step for the basic method. This compares very favorably to the idealized finite difference result for smooth curves, $\mathcal{O}(\frac{1}{\tau^3})$, which was derived in Section 2.4.2.

These are indeed formidable advantages for the new, spectral method over the usual finite difference approach. Even when the fast transform-based methods of the previous section are not used, large computational savings are typically observed.

To illustrate the performance improvement, consider the motion by mean curvature of the kidney-shaped region displayed in Figure 3.21. Using the new, spectral method[5] and a finite difference approach[6], we compare the area lost over a time $t = 0.0125$ with the exact answer, $0.0125 \times \pi = 0.0785398$ (see [51]). From Table 1, we see that the new, spectral method is adequate for finding solutions to within a 1% error. As we shall see in the next chapter, even more accurate results are practical using the transform methods

[5] A direct evaluation of the Fourier summations was carried out.

[6] The difference algorithm uses an adaptive τ-stepping method (see Section 2.4.2) on a uniform mesh. A multigrid technique [3] was used to solve the implicit equations which arose from a backward Euler time-stepping scheme.

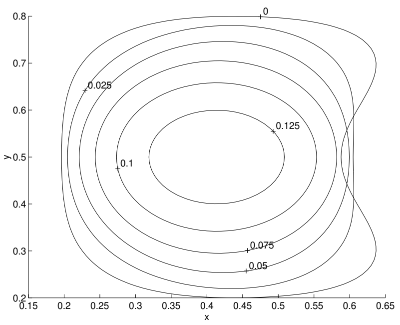


Figure 3.21: A Smooth Interface at Time, t

described in Section 3.5. The finite difference approach, however, is impractical when accurate solutions are sought (see Table 2).

Numerical tests for the problems described in the next chapter also found that the new, spectral method often requires less than 0.1% of the computational time of the usual finite difference approach. For this reason, the numerical studies in the following two chapters are carried out using the new, spectral method.

τ	h	*Error*	*Time*[7]
0.003125	2^{-9}	4%	0.4 s
0.00078125	2^{-11}	1%	8 s

Table 1. New, Spectral Method

Δt	Δx	*Error*	*Time*[7]
1×10^{-4}	$\frac{1}{128}$	4%	85 s
2×10^{-5}	$\frac{1}{512}$	3%	10341 s

Table 2. Finite Difference Discretization

[7]All timings were carried out on an HP735/100 workstation.

Chapter 4

Theoretical and Numerical Studies

In this chapter, the order of accuracy of the MBO-method is studied for a variety of two dimensional initial shapes which move according to mean curvature motion. In particular, theoretical and numerical studies are presented to show that the method produces a first order error in τ for smooth, two-phase problems. For three-phase problems, an expansion is derived to help explain why junction angles remain within $\mathcal{O}(\sqrt{\tau})$ of the correct configuration and numerical experiments are presented to show that the basic method demonstrates $\mathcal{O}(\sqrt{\tau})$ errors in actual computations. Numerical studies for initial corners and for singularities in the solution are also presented. Throughout this chapter, extrapolation in τ is used to produce higher order methods.

Our efforts fall short of a global convergence analysis. What we present instead is a combination of heuristics: Local error analysis coupled with numerical experiments which suggest that the derived local error orders may also hold globally.

4.1 Smooth Interfaces

In this section, we consider the application of the MBO-method to smooth curves. Specifically, we determine the local order of accuracy of the method and we propose an extrapolation to provide a higher order method.

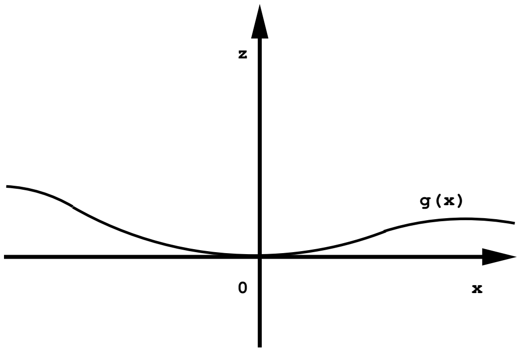


Figure 4.1: The Initial Interface

4.1.1 Truncation Error Analysis

We begin this chapter with a demonstration that the MBO-method is locally first order accurate in τ for smooth curves. Specifically, we show that the level set $F(x, z, \tau) = \frac{1}{2}$ for

$$\begin{aligned} \frac{\partial F}{\partial t} &= \Delta F, \\ F(x, z, 0) &= \begin{cases} 1 & \text{if } z > g(x) \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

where $g(x)$ is the initial interface, gives an $\mathcal{O}(\tau^2)$ approximation to the position of the front after a time, τ.

Begin by considering a smooth (4 times differentiable) interface which initially passes through the origin, tangent to the x-axis[1] as in Figure 4.1. By [45], the value of $F(x, z, t)$ along the z-axis is given by

$$F(0, z, t) = \frac{1}{2} - \frac{1}{\sqrt{4\pi t}} \int_0^z e^{-\frac{y^2}{4t}} dy + \frac{1}{4\pi t} \int_{-\infty}^{\infty} e^{-\frac{x^2}{4t}} \int_0^{g(x)} e^{-\frac{(z-y)^2}{4t}} dy\ dx.$$

Expanding two of the exponentials around 0, and replacing $g(x)$ by the first few terms

[1] For consistency with [45], the x-z plane is considered rather than the usual x-y plane.

of the Taylor series gives,

$$\begin{aligned}
F(0,z,t) &= \frac{1}{2} - \frac{1}{\sqrt{4\pi t}}\int_0^z \left(1-\frac{y^2}{4t}\right)dy + \\
&\quad \frac{1}{4\pi t}\int_{-\infty}^{\infty} e^{-\frac{x^2}{4t}} \int_0^{\frac{1}{2}g^{(2)}(0)x^2+\frac{1}{3!}g^{(3)}(0)x^3+\frac{1}{4!}g^{(4)}(0)x^4} \left[1-\frac{(z-y)^2}{4t}\right] dy\ dx + \text{h.o.t.}, \\
&= \frac{1}{2} - \frac{1}{\sqrt{4\pi t}}\left(z-\frac{z^3}{12t}\right) + \\
&\quad \frac{1}{4\pi t}\int_{-\infty}^{\infty} e^{-\frac{x^2}{4t}} \left[y + \frac{(z-y)^3}{12t}\right]_{y=0}^{y=\frac{1}{2}g^{(2)}(0)x^2+\frac{1}{3!}g^{(3)}(0)x^3+\frac{1}{4!}g^{(4)}(0)x^4} dx + \text{h.o.t.}, \\
&= \frac{1}{2} - \frac{1}{\sqrt{4\pi t}}\left(z-\frac{z^3}{12t}\right) + \\
&\quad \frac{1}{4\pi t}\int_{-\infty}^{\infty} e^{-\frac{x^2}{4t}} \left[y + \frac{-3z^2y+3zy^2-y^3}{12t}\right]_{y=0}^{y=\frac{1}{2}g^{(2)}(0)x^2+\frac{1}{3!}g^{(3)}(0)x^3+\frac{1}{4!}g^{(4)}(0)x^4} dx + \text{h.o.t.}
\end{aligned}$$

Noting that terms which are odd powers of x cancel in the integral, and writing $g^{(n)}$ for $g^{(n)}(0)$ we find

$$F(0,z,t) = \frac{1}{2} - \frac{1}{\sqrt{4\pi t}}\left(z-\frac{z^3}{12t}\right) +$$
$$\frac{1}{4\pi t}\int_{-\infty}^{\infty} e^{-\frac{x^2}{4t}} \left[\frac{1}{2}g^{(2)}x^2 + \frac{1}{4!}g^{(4)}x^4 + \frac{-\frac{3}{2}z^2g^{(2)}x^2 + 3z\left(\frac{1}{2}g^{(2)}x^2\right)^2 - \left(\frac{1}{2}g^{(2)}x^2\right)^3}{12t}\right] dx + \text{h.o.t.}$$

Applying integration by parts and the well-known identity,

$$\int_{-\infty}^{\infty} e^{-at^2} dt = \sqrt{\frac{\pi}{a}}$$

yields

$$\begin{aligned}
F(0,z,t) &= \frac{1}{2} - \frac{z}{2\sqrt{\pi t}} + \frac{z^3}{24t\sqrt{\pi t}} + \\
&\quad \frac{g^{(2)}}{2}\sqrt{\frac{t}{\pi}}\left(1-\frac{z^2}{4t}\right) + 6t\sqrt{\frac{t}{\pi}}\left[\frac{g^{(4)}}{4!} + \frac{z\left(g^{(2)}\right)^2}{16t}\right] - \frac{5\left(g^{(2)}\right)^3 t^{\frac{5}{2}}}{8t\sqrt{\pi}} + \text{h.o.t.}
\end{aligned} \tag{4.28}$$

To find where the interface intersects the z-axis, set $F(0,z,t) = \frac{1}{2}$ to obtain:

$$z = \frac{z^3}{12t} + g^{(2)}\left(t - \frac{z^2}{4}\right) + 12\left[\frac{g^{(4)}}{4!} + \frac{z\left(g^{(2)}\right)^2}{16t}\right]t^2 - \frac{5}{4}\left(g^{(2)}\right)^3 t^2 + \text{h.o.t.}$$

Substitution of the first order approximation of z into the right hand side gives a higher order estimate which simplifies to

$$z(t) = \frac{1}{12}\left(g^{(2)}\right)^3 t^2 + g^{(2)}t - \frac{1}{4}\left(g^{(2)}\right)^3 t^2 + \frac{1}{2}g^{(4)}t^2 + \frac{3}{4}\left(g^{(2)}\right)^3 t^2 - \frac{5}{4}\left(g^{(2)}\right)^3 t^2 + \mathcal{O}(t^3).$$

Thus,

$$z(\tau) = g^{(2)}\tau + \left[\frac{1}{2}g^{(4)} - \frac{2}{3}\left(g^{(2)}\right)^3\right]\tau^2 + \mathcal{O}(\tau^3). \tag{4.29}$$

Using a known result (see [54]) for the exact position, z^*,

$$z_t^* = \frac{z_{xx}^*}{1+(z_x^*)^2}$$

it is straightforward to show that

$$z^*(\tau) = g^{(2)}\tau + \left[\frac{1}{2}g^{(4)} - \left(g^{(2)}\right)^3\right]\tau^2 + \mathcal{O}(\tau^3). \tag{4.30}$$

Thus, the MBO-method is locally first order in τ for smooth curves.

4.1.2 Extrapolation

From Equations (4.29) and (4.30) we expect that extrapolation in τ can be used to produce higher order accurate results since the error,

$$\frac{1}{3}\left(g^{(2)}\right)^3\tau^2 + \mathcal{O}(\tau^3)$$

varies smoothly with τ and depends only on the local properties of the curve.

Assuming that the sharpened Fourier coefficients at time t are given by $\{c_{ij}(t)\}$, an extrapolated method for smooth curves may be obtained as follows:

Extrapolation I

1. Carry out one step of the MBO-method with a step size 2τ to obtain the sharpened coefficients, $c_{ij}^{2\tau}(t+2\tau)$.

2. Carry out two steps of the method with a step size τ to obtain sharpened coefficients, $c_{ij}^{\tau}(t+2\tau)$.

3. Set
$$c_{ij}(t+2\tau) = 2c_{ij}^{\tau}(t+2\tau) - c_{ij}^{2\tau}(t+2\tau)$$
to obtain the extrapolated coefficients. □

We hope to obtain second order accurate results in τ using this extrapolation. To understand why, suppose $M(\tau)$ produces an $\mathcal{O}(\tau)$ approximation to some unknown quantity M^*. Assuming the error for the approximation of $M(\tau)$ to M^* can be expressed as

$$M^* = M(\tau) + K\tau + \mathcal{O}(\tau^2)$$

where K is independent of τ, we find that the extrapolation

$$2M(\tau) - M(2\tau) = M^* + \mathcal{O}(\tau^2)$$

is second order accurate. See [32, 34, 37] for treatments of extrapolation for initial value problems.

The extrapolation process may be repeated, of course, as in Romberg's integration to obtain methods which we expect are even higher order accurate. See, for example, [15].

4.1.3 Numerical Experiments

From the previous two sections, we expect that the basic and extrapolated methods will give globally first and second order convergence rates, respectively. To experimentally verify these convergence estimates, we consider the motion by mean curvature of a shrinking ellipse with principle axes 0.8 and 0.6. (Similar results are obtained for the shapes of Figure 3.21 (cf. [60]) and Figure 3.14a.) Using the basic method and Extrapolation I, the area lost over a time $t = 0.025$ was compared to the exact answer[2] for several τ. The results from a number of experiments are given in Table 3, below.

τ	Basic Method		Extrapolation I	
	Error	*Conv. Rate*[3]	*Error*	*Conv. Rate*
0.00125	-8.31e-04	1.0	-4.05e-05	-
0.000625	-4.11e-04	1.0	7.71e-06	2.39
0.0003125	-2.05e-04	1.0	2.02e-06	1.93
0.00015625	-1.02e-04	1.0	4.97e-07	2.02

Table 3. Extrapolated Results for Two Phases

These results support the conclusion that the MBO-method is first order in τ and suggest that extrapolation can be used in conjunction with the new, spectral method to produce higher order results.

The extrapolated errors in Table 3 change sign (and the convergence rate fluctuates) because the first step of the method uses the characteristic set of the initial shape in the diffusive step, whereas later steps use extrapolated results which can take on values between -1 and 2 (see, e.g., Figure 4.2). Clearer estimates of the convergence rate are obtained by computing the errors that arise away from $t = 0$. For example, Table 4 gives the error in the area lost from time $t = 0.02$ to $t = 0.025$ for several values of τ.

[2]The exact result is easily derived using Equation (2.3).

[3]If the error for a step of size τ is E_τ, then we estimate the convergence rate as $\log_2\left(\left|\frac{E_{2\tau}}{E_\tau}\right|\right)$.

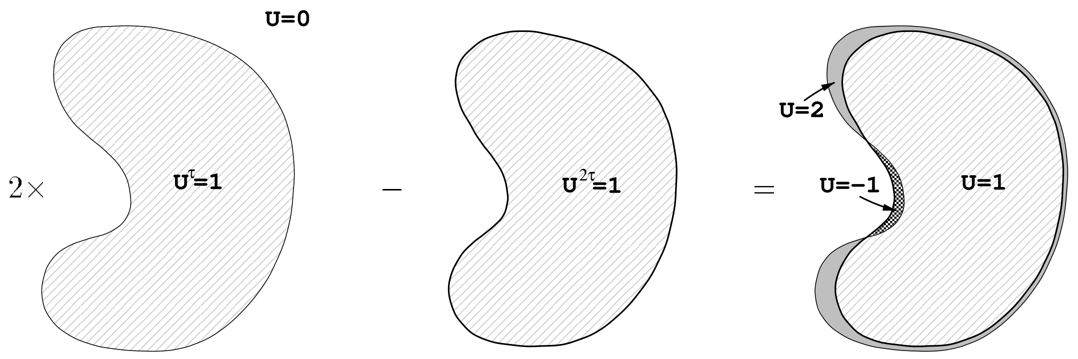


Figure 4.2: Extrapolated, Semi-Discrete Result

τ	*Error*	*Conv. Rate*
0.00125	9.66e-06	-
0.000625	2.28e-06	2.08
0.0003125	5.53e-07	2.04
0.00015625	1.34e-07	2.04

Table 4. Extrapolated Method for Two Phases

These results support the conjecture that the extrapolated method is second order in τ.

4.2 Nonsmooth Boundaries

In the previous section, we demonstrated that the MBO-method produces a first order error in τ for smooth curves. We now consider the application of the method to nonsmooth initial shapes. In order to determine the form of the error that arises for nonsmooth corners, we consider a fixed τ. To produce an optimized code, however, variable sized steps in τ should be considered.

Consider, for example, the motion by mean curvature of the initially nonsmooth interface given in Figure 4.3. Using the basic method and Extrapolation I, the area lost

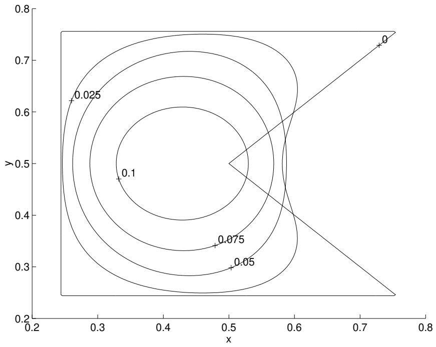


Figure 4.3: Initially Nonsmooth Interface at Times, t

over a time $t = 0.0125$ was compared to the exact answer[2] for several τ. The results over a number of experiments are given in Table 5, below.

τ	Basic Method		Extrapolation I	
	Error	*Conv. Rate*	*Error*	*Conv. Rate*
0.0015625	6.40e-03	0.71	-1.98e-03	1.16
0.00078125	3.75e-03	0.77	-8.94e-04	1.15
0.000390625	2.15e-03	0.80	-4.23e-04	1.08
0.0001953125	1.21e-03	0.83	-2.06e-04	1.04

Table 5. Basic and Extrapolated Results for a Nonsmooth Shape

These results suggest that the extrapolation is first order in τ. To obtain a clearer understanding of the form of these errors, let $\epsilon(\tau)$ be the error which arises for the basic method using a step size τ. Assuming that the extrapolated result is indeed first order, i.e.

$$2\epsilon(\tau) - \epsilon(2\tau) = c\tau + o(\tau)$$

for some constant c, we write

$$\frac{\epsilon(2\tau) - \epsilon(\tau)}{\tau} - \frac{\epsilon(\tau)}{\tau} \quad = \quad -c + o(1),$$

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}\tau}[\epsilon(\tau)] - \frac{\epsilon(\tau)}{\tau} &= -c + o(1), \\
\frac{\mathrm{d}}{\mathrm{d}\tau}\left(\frac{\epsilon(\tau)}{\tau}\right) &= -\frac{c}{\tau} + o\left(\frac{1}{\tau}\right).
\end{aligned}$$

Integrating with respect to τ, we find that

$$\begin{aligned}
\epsilon(\tau) &= -c\tau\log(\tau) + o(\tau\log(\tau)), \\
&= \mathcal{O}(\tau\log(\tau)).
\end{aligned}$$

Thus, we suspect that the basic method is order $\tau\log(\tau)$ for the case of nonsmooth initial corners[4].

To derive a higher order method for nonsmooth curves, we assume that the sharpened Fourier coefficients at time t are given by $\{c_{ij}\}$ and carry out Extrapolation I twice to eliminate the leading order error term. This repeated extrapolation produces the following method:

Extrapolation II

1. Carry out one step of the MBO-method with a step size 4τ to obtain the sharpened coefficients, $c_{ij}^{4\tau}(t+4\tau)$.

2. Carry out two steps of the MBO-method with a step size 2τ to obtain the sharpened coefficients, $c_{ij}^{2\tau}(t+4\tau)$.

3. Carry out four steps of the method with a step size τ to obtain sharpened coefficients, $c_{ij}^{\tau}(t+4\tau)$.

4. Set
$$c_{ij}(t+4\tau) = 4c_{ij}^{\tau}(t+4\tau) - 4c_{ij}^{2\tau}(t+4\tau) + c_{ij}^{4\tau}(t+4\tau)$$

[4]Note that if the order is indeed $\mathcal{O}(\tau\log(\tau))$ then the quantity measured as convergence rate gives $1+\log_2(1+(\log_2\tau)^{-1}) \to 1$ as $\tau \to 0$ which is consistent with the results in Table 5.

to obtain the extrapolated coefficients. □

Applying this extrapolated method to the nonsmooth problem yields the results given in Table 6.

τ	| *Error* |	*Conv. Rate*
0.0015625	4.91e-04	-
0.00078125	2.60e-04	0.91
0.000390625	7.46e-05	1.80
0.0001953125	1.89e-05	1.98

Table 6. Results for a Nonsmooth Shape Using Repeated Extrapolation

These results show that repeated extrapolation can be very effective, even when nonsmooth initial corners are present. Indeed, Table 6 suggests that an approximately second order method arises from the use of Extrapolation II.

Topological mergings and breakings in three dimensional or affine flows can also produce nonsmooth corners. For this reason, we expect no better than an $\mathcal{O}(\tau \log(\tau))$ error to arise in these situations. To obtain a deeper understanding of the form of these errors, further studies are needed since an additional dependency on nonlocal properties of the interface occurs.

4.3 Singularities in the Solution as Regions Disappear

To conclude our study of the MBO-method for two-phase problems, we consider the method's treatment of regions as they shrink to points and disappear under mean curvature motion. Similar to the case of nonsmooth corners, this motion causes condition (2.6) to be violated. For this reason, we suspect that an $\mathcal{O}(\tau \log(\tau))$ error may arise whenever the basic method is applied.

To test if an $\mathcal{O}(\tau \log(\tau))$ error occurs in practice, we consider the motion by mean curvature of a shrinking ellipse with principle axes 0.2 and 0.25. Using the basic method, the disappearance time T^τ was compared to the exact answer, $T^e = 0.025$, for several τ. An extrapolation in the disappearance time, $4T^\tau - 4T^{2\tau} + T^{4\tau}$, was also computed. The results for a number of experiments are given in Table 7, below.

τ	$T^\tau - T^e$	*Conv. Rate*	$(4T^\tau - 4T^{2\tau} + T^{4\tau}) - T^e$
0.005	-3.15e-03	-	1.49e-03
0.0025	-1.90e-03	0.73	-3.16e-04
0.00125	-1.12e-03	0.76	-1.85e-05
0.000625	-6.42e-04	0.80	8.34e-06

Table 7. Basic and Extrapolated Errors for a Shrinking Ellipse

These results show a slow tendency upwards to first order for the basic method and indicate that extrapolation can be very effective as phase regions disappear. (Unfortunately, we were unable to determine a clear estimate of the convergence rate for the extrapolation.) Similar to the case of nonsmooth corners, these results are suggestive of an order $\tau \log(\tau)$ error for the basic method.

4.4 Junctions in Two Dimensions

In this section, we consider the application of the MBO-method to triple junctions in two dimensions. Specifically, we present numerical experiments to estimate the order of accuracy of the basic method and propose an extrapolated method for improved accuracy. We also derive asymptotic results which explain the stability of junction angles and suggest a source of the $\mathcal{O}(\sqrt{\tau})$ error which arises in numerical experiments.

4.4.1 Error Analysis

Under mean curvature motion, two dimensional triple junctions form a stable $\frac{2\pi}{3}$-$\frac{2\pi}{3}$-$\frac{2\pi}{3}$ angle configuration (see, e.g., [13]). In this subsection, we show that each step of the MBO-method produces an $\mathcal{O}(\sqrt{\tau})$ error in the junction angles which is rapidly dissipated during subsequent steps.

We now derive an expansion for the angles of a two dimensional triple junction after one step of the MBO-method. We shall assume that each angle approximates $\frac{2}{3}\pi$, initially.

The Initial Junction

We begin by orienting a polar coordinate system so that the junction angle furthest from $\frac{2}{3}\pi$ is centered about $\theta = 0$. We denote the initial interfaces by ℓ_0, ℓ_1 and ℓ_2 and the initial regions by R_0, R_1 and R_2 as in Figure 4.4.

To represent the small deviations from the $\frac{2\pi}{3}$-$\frac{2\pi}{3}$-$\frac{2\pi}{3}$ junction configuration we define

$$\begin{aligned} \epsilon &= \angle \ell_0 \ell_1 - \frac{2\pi}{3}, \\ c\epsilon &= \angle \ell_1 \ell_2 - \frac{2\pi}{3} \end{aligned}$$

where $\angle \ell_i \ell_j$ is the angle between ℓ_i and ℓ_j. Since $\angle \ell_0 \ell_1 + \angle \ell_1 \ell_2 + \angle \ell_2 \ell_0 = 2\pi$ it is easy to see that

$$\begin{aligned} |\epsilon| = \left| \angle \ell_0 \ell_1 - \frac{2\pi}{3} \right| &\geq \left| \angle \ell_1 \ell_2 - \frac{2\pi}{3} \right| = |c||\epsilon|, \\ |\epsilon| = \left| \angle \ell_0 \ell_1 - \frac{2\pi}{3} \right| &\geq \left| \angle \ell_2 \ell_0 - \frac{2\pi}{3} \right| = |\epsilon||1+c|. \end{aligned}$$

Hence, the constant c satisfies $-1 \leq c \leq 0$.

In order to carry out our expansions, we want an expression for each interface,

$$\ell_i = \{(r, \theta_{\ell_i}(r)) : r \geq 0\}$$

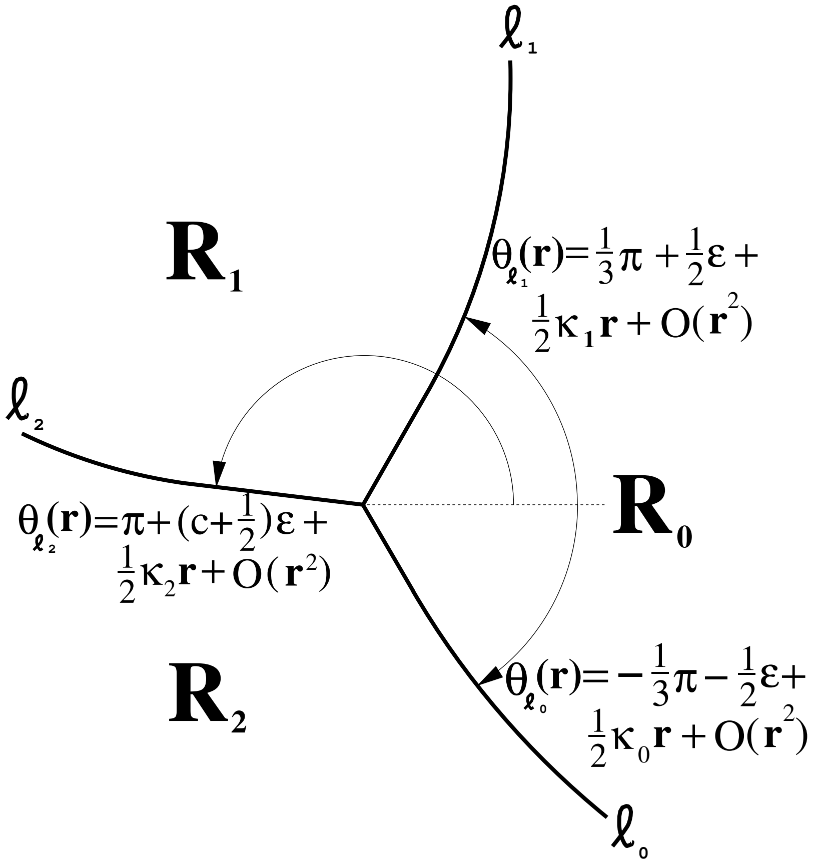


Figure 4.4: The Initial Junction

for some function, $\theta_{\ell_i}(r)$. Using the above definitions it is straightforward to show that

$$\begin{aligned}
\theta_{\ell_0}(r) &= -\frac{1}{3}\pi - \frac{1}{2}\epsilon + \frac{1}{2}\kappa_0 r + \mathcal{O}(r^2), \\
\theta_{\ell_1}(r) &= \frac{1}{3}\pi + \frac{1}{2}\epsilon + \frac{1}{2}\kappa_1 r + \mathcal{O}(r^2), \\
\theta_{\ell_2}(r) &= \pi + \left(c + \frac{1}{2}\right)\epsilon + \frac{1}{2}\kappa_2 r + \mathcal{O}(r^2)
\end{aligned}$$

where κ_i is the curvature of line ℓ_i at the origin.

Approximation of U_i

We now want to estimate $\vec{U} = (U_0, U_1, U_2)$ after a time, τ. At $t = 0$, we know that

$$U_i(r,\theta,0) = \begin{cases} 1 & \text{if } (r,\theta) \in R_i \\ 0 & \text{otherwise,} \end{cases}$$

for $0 \leq i \leq 2$.

The Green's function representation of $U_0(r,\theta,\tau)$ gives

$$\begin{aligned} U_0(r,\theta,\tau) &= \frac{1}{4\pi\tau}\int_0^\infty \int_{\theta_{\ell_0}(R)}^{\theta_{\ell_1}(R)} \exp\left(-\frac{[R\cos(\phi) - r\cos(\theta)]^2 + [R\sin(\phi) - r\sin(\theta)]^2}{4\,\tau}\right) R\, d\phi\, dR, \\ &= \frac{1}{4\pi\tau}\exp\left(-\frac{r^2}{4\tau}\right)\int_0^\infty \exp\left(-\frac{R^2}{4\tau}\right)\int_{\theta_{\ell_0}(R)}^{\theta_{\ell_1}(R)} \exp\left(\frac{r R\cos(\phi-\theta)}{2\,\tau}\right) R\, d\phi\, dR. \end{aligned}$$

Replacing the exponential in the inner integral by its series and integrating term by term yields,

$$\begin{aligned} U_0(r,\theta,\tau) &= \frac{1}{4\pi\tau}\exp\left(-\frac{r^2}{4\tau}\right)\int_0^\infty \sum_{n=0}^{\infty}\left[R\frac{1}{n!}\left(\frac{rR}{2\,\tau}\right)^n \exp\left(-\frac{R^2}{4\tau}\right)\int_{\theta_{\ell_0}(R)}^{\theta_{\ell_1}(R)} \cos^n(\phi-\theta)\, d\phi\right] dR, \\ &= \frac{1}{4\pi\tau}\exp\left(-\frac{r^2}{4\tau}\right)\int_0^\infty R\exp\left(-\frac{R^2}{4\tau}\right) \qquad (4.31) \\ &\quad \times\Big(\theta_{\ell_1}(R) - \theta_{\ell_0}(R) + \frac{rR}{2\tau}\left[\sin(\theta_{\ell_1}(R)-\theta) - \sin(\theta_{\ell_0}(R)-\theta)\right] \\ &\quad + \frac{1}{4}\left(\frac{rR}{2\tau}\right)^2\left[\theta_{\ell_1}(R) - \theta_{\ell_0}(R) + \frac{1}{2}\sin\left(2\left(\theta_{\ell_1}(R)-\theta\right)\right) - \frac{1}{2}\sin\left(2\left(\theta_{\ell_0}(R)-\theta\right)\right)\right]\Big)\, dR \\ &\quad + E(r,\theta,\tau) \end{aligned}$$

where

$$\begin{aligned} E(r,\theta,\tau) &= \sum_{n=3}^{\infty} E_n(r,\theta,\tau), \\ E_n(r,\theta,\tau) &= \frac{1}{4\pi\tau}\exp\left(-\frac{r^2}{4\tau}\right)\int_0^\infty \frac{R}{n!}\left(\frac{rR}{2\,\tau}\right)^n \exp\left(-\frac{R^2}{4\tau}\right)\int_{\theta_{\ell_0}(R)}^{\theta_{\ell_1}(R)} \cos^n(\phi-\theta)\, d\phi\, dR. \end{aligned}$$

We seek an estimate of $E_n(r,\theta,\tau)$. Noting that

$$\int_{\theta_{\ell_0}(R)}^{\theta_{\ell_1}(R)} \cos^n(\phi-\theta)\, d\phi < 2\pi,$$

it is clear that

$$\begin{aligned}
|E_n(r,\theta,\tau)| &\le \frac{1}{2\tau}\exp\left(-\frac{r^2}{4\tau}\right)\int_0^\infty \frac{R}{n!}\left(\frac{rR}{2\,\tau}\right)^n \exp\left(-\frac{R^2}{4\tau}\right)\, dR,\\
&= \frac{2}{n!}\left(\frac{r}{\sqrt{\tau}}\right)^n \exp\left(-\frac{r^2}{4\tau}\right)\int_0^\infty \xi^{n+1}\exp\left(-\xi^2\right)\, d\xi,\\
&\le \left(\frac{r}{\sqrt{\tau}}\right)^n \exp\left(-\frac{r^2}{4\tau}\right),\\
&\le \left(\frac{r}{\sqrt{\tau}}\right)^n.
\end{aligned}$$

Assuming $\frac{r}{\sqrt{\tau}} \ll 1$, we find

$$E(r,\theta,\tau) = \mathcal{O}\left(\left(\frac{r}{\sqrt{\tau}}\right)^3\right).$$

Similar to the above result, the second and higher order terms of $\theta_{\ell_i}(R)$ make an $\mathcal{O}\left(\tau + \frac{\epsilon r^2}{\tau} + \left(\frac{r}{\sqrt{\tau}}\right)^3\right)$ contribution to $U_0(r,\theta,\tau)$ in Equation (4.31). Thus,

$$\begin{aligned}
U_0(r,\theta,\tau) &= \frac{1}{4\pi\tau}\exp\left(-\frac{r^2}{4\tau}\right)\int_0^\infty R\exp\left(-\frac{R^2}{4\tau}\right)\left(\frac{2}{3}\pi + \epsilon + \frac{1}{2}(\kappa_1-\kappa_0)R\right.\\
&\quad + \frac{rR}{2\tau}\left[\sin\left(\frac{1}{3}\pi + \frac{1}{2}\epsilon + \frac{1}{2}\kappa_1 R - \theta\right) - \sin\left(-\frac{1}{3}\pi - \frac{1}{2}\epsilon + \frac{1}{2}\kappa_0 R - \theta\right)\right]\\
&\quad \left. + \frac{1}{4}\left(\frac{rR}{2\tau}\right)^2\left[\frac{2}{3}\pi + \frac{1}{2}\sin\left(\frac{2}{3}\pi - 2\theta\right) - \frac{1}{2}\sin\left(-\frac{2}{3}\pi - 2\theta\right)\right]\right)\, dR\\
&\quad + \mathcal{O}\left(\tau + \frac{\epsilon r^2}{\tau} + \left(\frac{r}{\sqrt{\tau}}\right)^3\right).
\end{aligned}$$

We now expand the integrand (using Maple [18]) and apply integration by parts with the well-known identity

$$\int_0^\infty \exp\left(-\frac{R^2}{a}\right)\, dR = \frac{1}{2}\sqrt{a\pi}$$

to obtain

$$U_0(r,\theta,\tau) = \frac{1}{3} + \frac{1}{2\pi}\epsilon + \frac{1}{4}(\kappa_1-\kappa_0)\sqrt{\frac{\tau}{\pi}} + \frac{r\sqrt{3}\cos(\theta)}{4\sqrt{\tau\pi}}$$

$$+\frac{1}{4\pi}(\kappa_1-\kappa_0)r\cos(\theta)+\frac{\sqrt{3}}{4\pi}(\kappa_1+\kappa_0)r\sin(\theta)+\frac{\epsilon}{8\sqrt{\pi\tau}}r\cos(\theta)$$
$$+\frac{\sqrt{3}}{16\pi\tau}\left(\cos^2(\theta)-\sin^2(\theta)\right)r^2+\mathcal{O}\left(\tau+\frac{\epsilon r^2}{\tau}+\left(\frac{r}{\sqrt{\tau}}\right)^3\right).$$

Re-writing in Cartesian coordinates yields

$$\begin{aligned} U_0(x,y,\tau) &= \frac{1}{3}+\frac{1}{2\pi}\epsilon+\frac{1}{4}(\kappa_1-\kappa_0)\sqrt{\frac{\tau}{\pi}}+\frac{\sqrt{3}}{4\sqrt{\tau\pi}}x \\ &\quad +\frac{1}{4\pi}(\kappa_1-\kappa_0)x+\frac{\sqrt{3}}{4\pi}(\kappa_1+\kappa_0)y+\frac{\epsilon}{8\sqrt{\pi\tau}}x+\frac{\sqrt{3}}{16\pi\tau}\left(x^2-y^2\right) \\ &\quad +\mathcal{O}\left(\tau+\frac{\epsilon}{\tau}\left(x^2+y^2\right)+\left(x^2/\tau+y^2/\tau\right)^{\frac{3}{2}}\right). \end{aligned} \tag{4.32}$$

Expansions for U_1 and U_2 may be obtained via rotations of (4.32) to give

$$\begin{aligned} U_1(x,y,\tau) &= \frac{1}{3}+\frac{1}{2\pi}c\epsilon+\frac{1}{4}(\kappa_2-\kappa_1)\sqrt{\frac{\tau}{\pi}}+\frac{3y-\sqrt{3}x}{8\sqrt{\tau\pi}}-\frac{1}{4\pi}(2\kappa_2+\kappa_1)x-\frac{\sqrt{3}\kappa_1}{4\pi}y \\ &\quad -\frac{3+4c}{16\sqrt{\pi\tau}}\epsilon x-\frac{\sqrt{3}}{16\sqrt{\pi\tau}}\epsilon y-\frac{3}{16\pi\tau}xy+\frac{\sqrt{3}}{32\pi\tau}\left(y^2-x^2\right) \\ &\quad +\mathcal{O}\left(\tau+\frac{\epsilon}{\tau}\left(x^2+y^2\right)+\left(x^2/\tau+y^2/\tau\right)^{\frac{3}{2}}\right). \\ U_2(x,y,\tau) &= 1-U_0(x,y,\tau)-U_1(x,y,\tau). \end{aligned}$$

Angle Expansions

We now seek expansions for the angle configuration of the junction after a time τ.

Begin by letting ℓ_0^τ, ℓ_1^τ and ℓ_2^τ be the MBO-approximations to the branches of the triple junction after a time τ. To approximate the angle between ℓ_0^τ and ℓ_1^τ, we require the location of the triple junction at time τ. This can be found by solving the system

$$\begin{cases} U_0(x_\tau,y_\tau,\tau)=U_1(x_\tau,y_\tau,\tau), \\ U_0(x_\tau,y_\tau,\tau)=U_2(x_\tau,y_\tau,\tau) \end{cases}$$

for (x_τ, y_τ) to give

$$x_\tau = -2\sqrt{\frac{\tau}{3\pi}}\epsilon-\frac{1}{\sqrt{3}}(\kappa_1-\kappa_0)\tau+\mathcal{O}(\epsilon\tau)+\mathcal{O}\left(\tau^{\frac{3}{2}}\right),$$

$$y_\tau = -\frac{2}{3}(2c+1)\sqrt{\frac{\tau}{\pi}}\epsilon + \frac{1}{3}(\kappa_0 - 2\kappa_2 + \kappa_1)\tau + \mathcal{O}(\epsilon\tau) + \mathcal{O}\left(\tau^{\frac{3}{2}}\right).$$

Our next task is to find the slope of ℓ_0^τ at the triple junction. Since ℓ_0^τ is given by

$$\{(x,y) : U_0(x,y,\tau) = U_2(x,y,\tau)\},$$

the slope of ℓ_0^τ is given implicitly by

$$D_x\left[U_0(x,y,\tau) - U_2(x,y,\tau)\right] = 0.$$

Solving for the slope at the triple junction, (x_τ, y_τ), yields

$$m_0 = -\sqrt{3} + \frac{2}{3}\left(c - 1 + (c+2)\frac{2}{\pi}\right)\epsilon + 2\left(2\kappa_0 + \kappa_1 + \kappa_2\right)\sqrt{\frac{\tau}{\pi}} + \mathcal{O}(\tau) + \mathcal{O}\left(\epsilon^2\right).$$

Similarly, the slope of ℓ_1^τ at the triple junction is given by

$$m_1 = \sqrt{3} + \frac{2}{3}\left(c + 2 + (c-1)\frac{2}{\pi}\right)\epsilon + 2(\kappa_0 + 2\kappa_1 + \kappa_2)\sqrt{\frac{\tau}{\pi}} + \mathcal{O}(\tau) + \mathcal{O}\left(\epsilon^2\right).$$

Thus, the value of the desired angle is given by

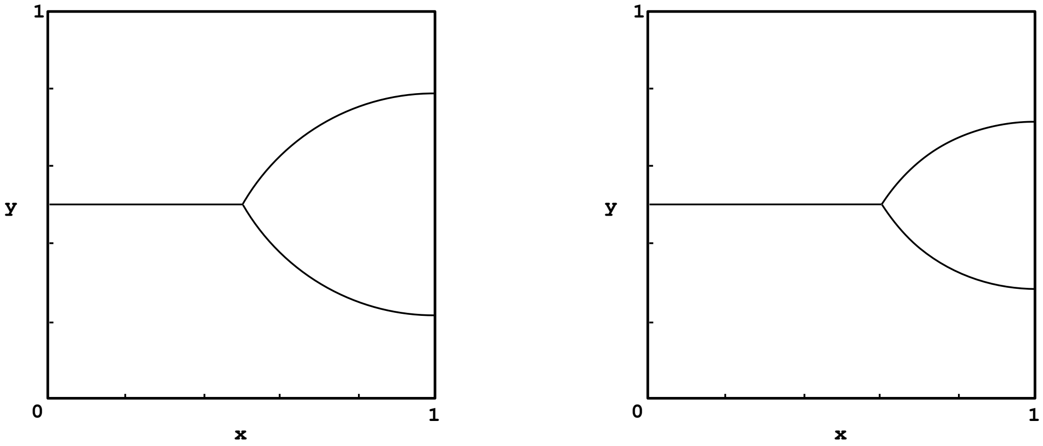
$$\begin{aligned} \angle\ell_0^\tau\ell_1^\tau &= \pi - \arctan\left(\frac{m_1 - m_0}{1 + m_0 m_1}\right), \\ &= \frac{2}{3}\pi + \left(\frac{1}{2} - \frac{1}{\pi}\right)\epsilon + \frac{1}{2}\left(\kappa_1 - \kappa_0\right)\sqrt{\frac{\tau}{\pi}} + \mathcal{O}(\tau) + \mathcal{O}\left(\epsilon^2\right). \end{aligned} \tag{4.33}$$

Similarly,

$$\begin{aligned} \angle\ell_1^\tau\ell_2^\tau &= \frac{2}{3}\pi + \left(\frac{1}{2} - \frac{1}{\pi}\right)c\epsilon + \frac{1}{2}\left(\kappa_2 - \kappa_1\right)\sqrt{\frac{\tau}{\pi}} + \mathcal{O}(\tau) + \mathcal{O}\left(\epsilon^2\right), \\ \angle\ell_0^\tau\ell_2^\tau &= \frac{2}{3}\pi - \left(\frac{1}{2} - \frac{1}{\pi}\right)(1+c)\epsilon + \frac{1}{2}\left(\kappa_0 - \kappa_2\right)\sqrt{\frac{\tau}{\pi}} + \mathcal{O}(\tau) + \mathcal{O}\left(\epsilon^2\right). \end{aligned}$$

Thus, each step of the MBO-method produces an $\mathcal{O}(\sqrt{\tau})$ error in the junction angles which is rapidly dissipated during subsequent steps. Summing up such contributions[5] over many τ steps, we expect to obtain a rapidly converging geometric sum which gives rise to an $\mathcal{O}(\sqrt{\tau})$ error in total. This is an interesting result because it gives an explanation for the stability of junction angles and suggests a source of the $\mathcal{O}(\sqrt{\tau})$ error which arises in numerical experiments (see next section).

[5]This summation step is non-rigorous because it assumes, among other things, that κ_0, κ_1 and κ_2 are bounded independent of τ.



$t = 0.0$ $t = 0.04$

Figure 4.5: A Smooth Three-Phase Problem

4.4.2 Numerical Experiments

We have just seen that the MBO-method is expected to produce an $\mathcal{O}(\sqrt{\tau})$ error in the angle configuration of junctions. For this reason, we suspect that the basic method may produce an $\mathcal{O}(\sqrt{\tau})$ error when junctions are present.

To test if this is indeed the case, we consider the motion by mean curvature of the three-phase problem given in Figure 4.5. Using the basic method, the area lost for the central region, A^τ, over a time $t = 0.04$ was compared to the exact answer[6], 0.0837758, for several τ. Because an $\mathcal{O}(\sqrt{\tau})$ error seems plausible from our asymptotic results, an extrapolation in the area, $\frac{1}{\sqrt{2}-1}\left(\sqrt{2}A^\tau - A^{2\tau}\right)$, was also computed to eliminate the conjectured leading order error term. The results for a number of experiments are given in Table 8, below.

[6]Applying the Von Neumann-Mullins parabolic law [51], gives us that the area of the central phase obeys

$$\frac{\mathrm{d}A}{\mathrm{d}t} = \frac{1}{2} \times \frac{\pi}{3}(2-6).$$

τ	*Error in* A^τ	*Conv. Rate*	*Error in* $\frac{1}{\sqrt{2}-1}\left(\sqrt{2}A^\tau - A^{2\tau}\right)$	*Conv. Rate*
0.00625	2.60e-03	0.67	-1.08e-03	1.33
0.003125	1.70e-03	0.61	-4.63e-04	1.22
0.0015625	1.14e-03	0.57	-2.12e-04	1.13
0.00078125	7.79e-04	0.55	-1.00e-04	1.08
0.000390625	5.37e-04	0.54	-4.86e-05	1.05

Table 8. Basic MBO-method for Three Phases

These results support the conjecture that the MBO-method is $\mathcal{O}(\sqrt{\tau})$ for the case of junctions and suggest that extrapolation can be used in conjunction with the new, spectral method to produce higher order estimates of quantities of interest such as phase areas.

Accurate, extrapolated estimates of the disappearance time of the smallest phase for a more complicated three-phase problem (see Figure 4.6) have also been determined in this manner [60]. Using the basic method, the disappearance time, T^τ, was compared to an estimate of the exact answer[7] for several τ. An extrapolation in the disappearance time, $\frac{1}{\sqrt{2}-1}(\sqrt{2}T^\tau - T^{2\tau})$, was also computed. The results for a number of experiments are reported in Table 9, below.

τ	*Error in* T^τ	*Conv. Rate*	*Error in* $\frac{1}{\sqrt{2}-1}\left(\sqrt{2}T^\tau - T^{2\tau}\right)$	*Conv. Rate*
0.04	0.16435	-	0.07782	-
0.02	0.12103	0.44	0.01645	2.24
0.01	0.08802	0.46	0.00833	1.38
0.005	0.06354	0.47	0.00444	0.91

Table 9. Results for the Disappearance Time of a Phase Region

[7] This result, $T = 0.33051$, was obtained using Brian Wetton's front tracking code; see [14].

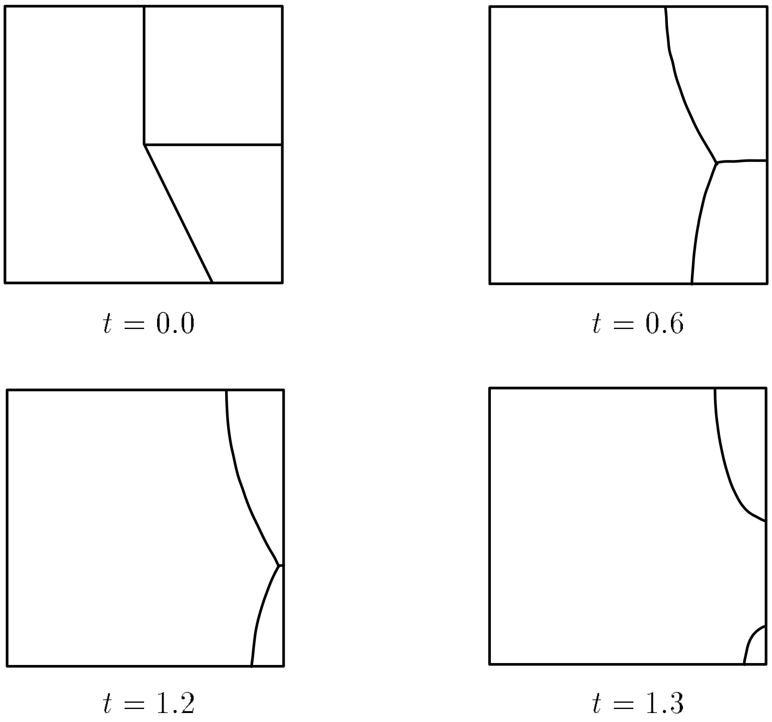


Figure 4.6: Evolution of a Junction Through a Singularity

These results are suggestive of an $\mathcal{O}(\sqrt{\tau})$ or $\mathcal{O}(\sqrt{\tau}\log(\tau))$ error for the basic method. We also find that a marked improvement in the error occurs when extrapolation is used.

An extrapolated method for approximating the entire interface is also sought. Assuming that the sharpened Fourier coefficients at time t are given by $\{c_{ij}(t)\}$, an extrapolated method for junctions may be obtained as follows:

Extrapolation III

1. Carry out one step of the MBO-method with a step size 2τ to obtain the sharpened coefficients, $c_{ij}^{2\tau}(t+2\tau)$.

2. Carry out two steps of the method with a step size τ to obtain sharpened coefficients, $c_{ij}^{\tau}(t+2\tau)$.

3. Set
$$c_{ij}(t+2\tau) = \frac{\sqrt{2}c_{ij}^{\tau}(t+2\tau) - c_{ij}^{2\tau}(t+2\tau)}{\sqrt{2}-1}$$
to obtain the extrapolated coefficients. □

Using this extrapolated method, approximations to the three-phase problem given in Figure 4.5 were computed. A comparison of the exact area change for the central region with the computed values yields the results given in Table 9.

τ	*Error*	*Conv. Rate*
0.00625	-1.10e-03	1.25
0.003125	-5.07e-04	1.11
0.0015625	-2.56e-04	0.98
0.00078125	-1.38e-04	0.89
0.000390625	-7.76e-05	0.83

Table 10. Extrapolated Method for Three Phases

Based on these results, it is not clear if the extrapolated algorithm possesses a higher order of accuracy than the basic method. Nonetheless, the extrapolated method may be of practical use since it typically results in a marked improvement in the error (cf. Tables 8 and 10).

4.5 Summary

Based on a number of two dimensional studies, we have determined the dominant local error term of the MBO-method for certain classes of problems and have suggested extrapolations for improved accuracy. We have observed corresponding behaviour of the error in selected actual computations. These results are summarized in Table 11 below.

Type of Interface	*Conjectured Error Order*	*Extrapolation Choice*
Two Phases:		
Smooth	τ	Extrapolation I
Nonsmooth	$\tau\log(\tau)$	Extrapolation II
Singularities	$\tau\log(\tau)$	Extrapolation II
Junctions	$\sqrt{\tau}$	Extrapolation III

Table 11. Summary of Theoretical and Numerical Studies in 2-D

For the remainder of this thesis, we focus on the approximation of three dimensional and nonlocal models of curvature-dependent motion.

Chapter 5

Numerical Experiments and Visualization

In this chapter, we apply the MBO-method to a number of three dimensional problems. In particular, we evolve surfaces with junctions according to mean curvature motion and visualize the results. An extension of the MBO-method to a nonlocal model that preserves phase areas is also proposed and studied.

To obtain results at an affordable cost, all problems are solved using the new, spectral method rather than with a finite difference discretization.

5.1 Three Dimensional, Two-Phase Problems

We begin this chapter by considering the motion by mean curvature of two-phase problems. Results developed here will also be used in the next section for the visualization of three dimensional junctions.

5.1.1 Visualization

To obtain a clearer understanding of the motion by mean curvature of surfaces, we seek a method for visualizing our results. This section describes a simple approach for generating movies of evolving shapes using Matlab [46]. See [6, 44] for discussions on more advanced visualization techniques.

Our principal task is to construct each frame of the movie. A relatively straightforward approach for generating surfaces with diffuse, ambient and specular lighting effects is to use Matlab's *surfl*(X, Y, Z) command [46]. This command produces a surface by interpolating between the points given by the matrices X, Y, and Z. Although it is possible to display small portions of a surface, we prefer to use larger segments. This avoids the shading variations that occasionally arise between segments when the full surface is displayed using a number of smaller fragments.

Certain smooth shapes are easy to represent as an $m \times m$ matrix. Suppose, for example, there is a point Q in the region which has a direct line of sight to the entire surface (e.g., point Q in Figure 5.1, but not point $\tilde{Q}$). Then, the following algorithm can be used to represent the shape:

Visualization I

Select a Q satisfying the "line of sight" property listed above.

(Currently Q is user selected.)

For $0 \le i, j \le m-1$,

Set $P_{ij} = (X_{ij}, Y_{ij}, Z_{ij})$ equal to the intersection of the surface with the line ℓ, where ℓ passes through Q and has an azimuth or horizontal rotation of $\frac{2\pi i}{m-1}$ and a vertical elevation of $\frac{2\pi j}{m-1} - \pi$.

This approach has been used to represent a variety of smooth surfaces (e.g., Figure 5.2). More complicated shapes have also been considered by dividing the shape into subregions and treating each separately (e.g., Figure 5.3).

Having constructed an appropriate $m \times m$ matrix, we need only call *surfl()* and specify a shading model to display the surface. Matlab provides for piecewise constant and Gouraud (piecewise bilinear) shading. For the surfaces we have considered, superior results arose from Gouraud shading (see, e.g., Figure 5.4).

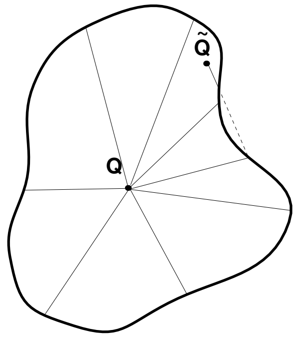


Figure 5.1: From Q the Entire Curve is Visible

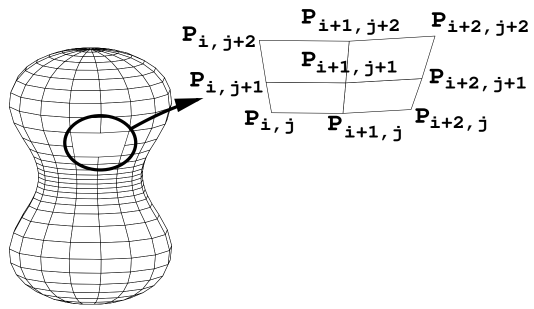


Figure 5.2: A Matrix Representation of the Surface

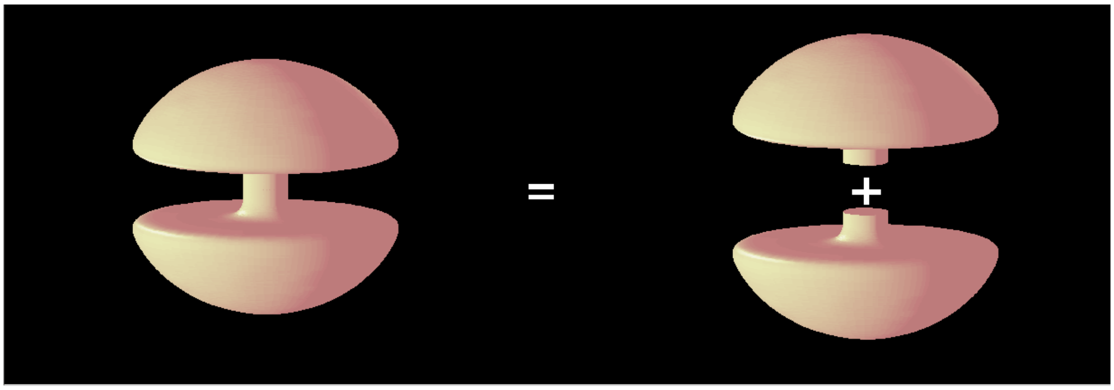


Figure 5.3: Splitting a Shape into Easily Parameterized Portions

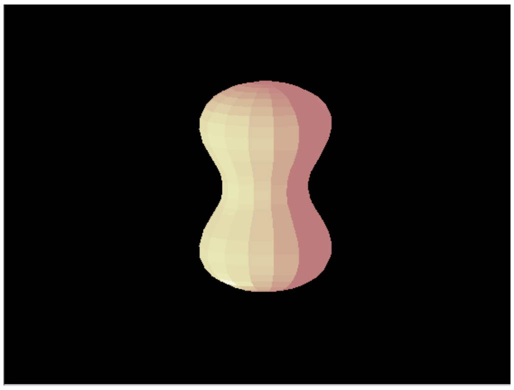


Fig. 5.4a. Piecewise Constant Shading

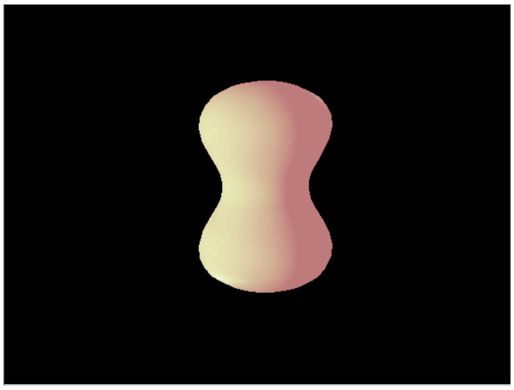


Fig. 5.4b. Gouraud Shading

Figure 5.4: Piecewise Constant and Gouraud Shading of the Surface

5.1.2 Numerical Experiments

We now report on several experiments for two-phase mean curvature motion. Throughout this section, a piecewise linear fit to the interface is used (see Section 3.3.2) with a finest cell width of $h = \frac{1}{128}$.

Collapsing Sphere

From the previous chapter, we saw that the basic MBO-method was first order in τ for smooth, two dimensional problems. To test if this result also holds in three dimensions, we consider the motion by mean curvature of a collapsing sphere with initial radius 0.4. Using the new, spectral discretization of the MBO-method the volume lost over a time $t = 0.03$ was compared to the exact answer, 0.16π, for several τ. The results for a number of experiments are given in Table 11, below.

τ	*Error*	*Conv. Rate*
0.01	0.0600	-
0.005	0.0258	1.22
0.0025	0.0119	1.12
0.00125	0.00582	1.03

Table 11. The Basic MBO-Method for the Shrinking Sphere

These results suggest that the basic MBO-method is first order in τ for smooth surfaces without junctions.

Thin-Stemmed Barbell

Examples involving topological changes are also naturally handled by the method. For example, Figure 5.5 displays the motion of a thin-stemmed barbell using a step size,

$\tau = 0.0004$. From these images, it is clear that the center handle pinches off to form two pieces. As expected from [39], these convex shapes become nearly spherical as they disappear.

Thick-Stemmed Barbell

A wider stem can produce a qualitatively different motion. For example, Figure 5.6 displays the motion of a thick-stemmed barbell using a step size, $\tau = 0.0005$. From these images, we see that no topological changes arise, and that the shape eventually becomes ellipsoidal and more spherical as it disappears.

5.2 Junctions in Three Dimensions

In this section, we consider the motion by mean curvature of surfaces with junctions. Specifically, we report on some numerical experiments and give a visualization method for certain multiple phase problems.

5.2.1 Visualization

The visualization of multiple phases can also be carried out in a straightforward manner for certain problems. Suppose, for example, that a clear phase surrounds several opaque regions. Then the following algorithm can be used to display the visible surfaces:

Visualization II

For each opaque phase, $0 \leq j \leq m$,

Draw the surface $U_j(x, y, z) = U_{\text{clear}}(x, y, z)$ before sharpening
using Matlab's *surfl()* command and Visualization I.

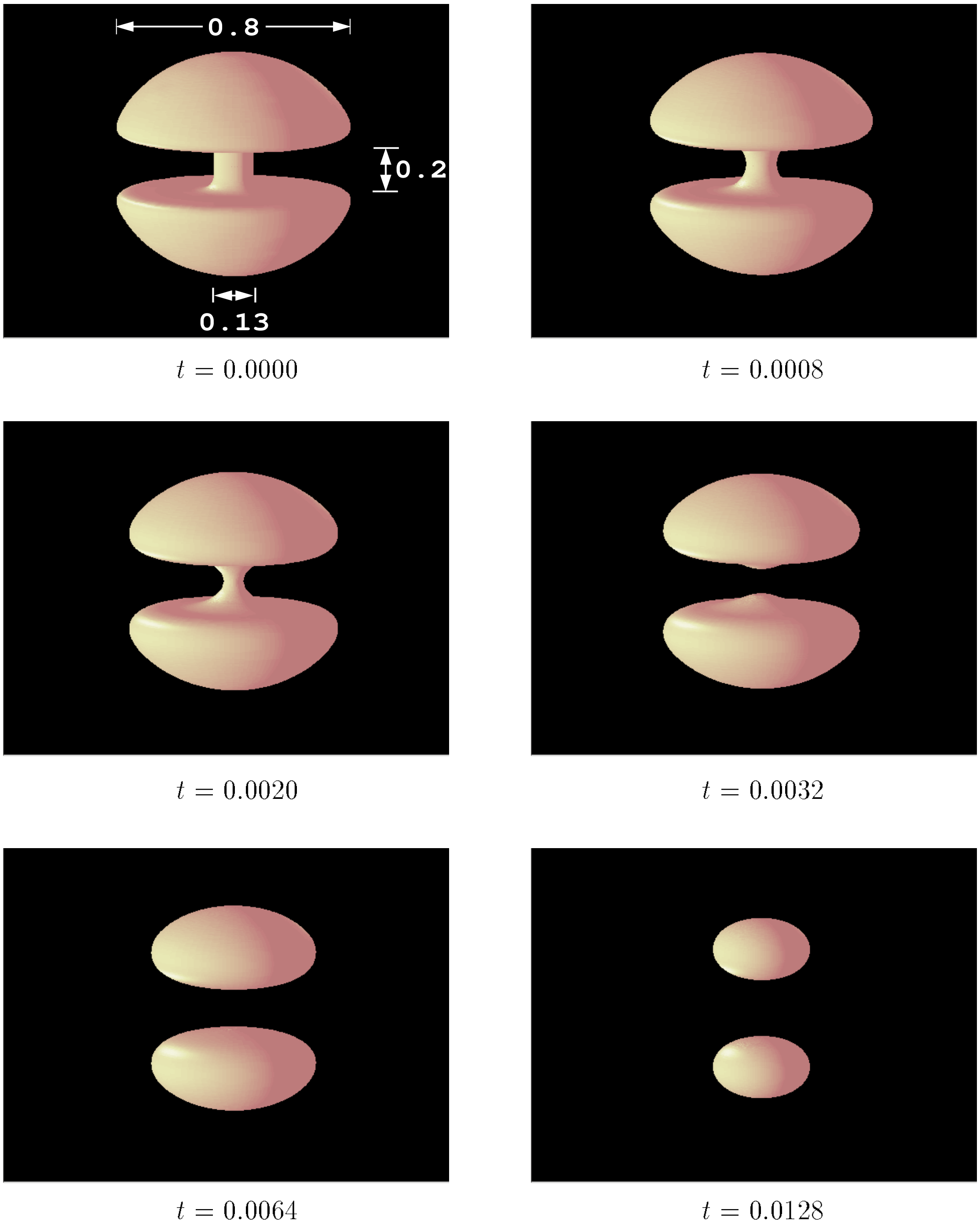


Figure 5.5: Thin-Stemmed Barbell Moving by Mean Curvature Motion

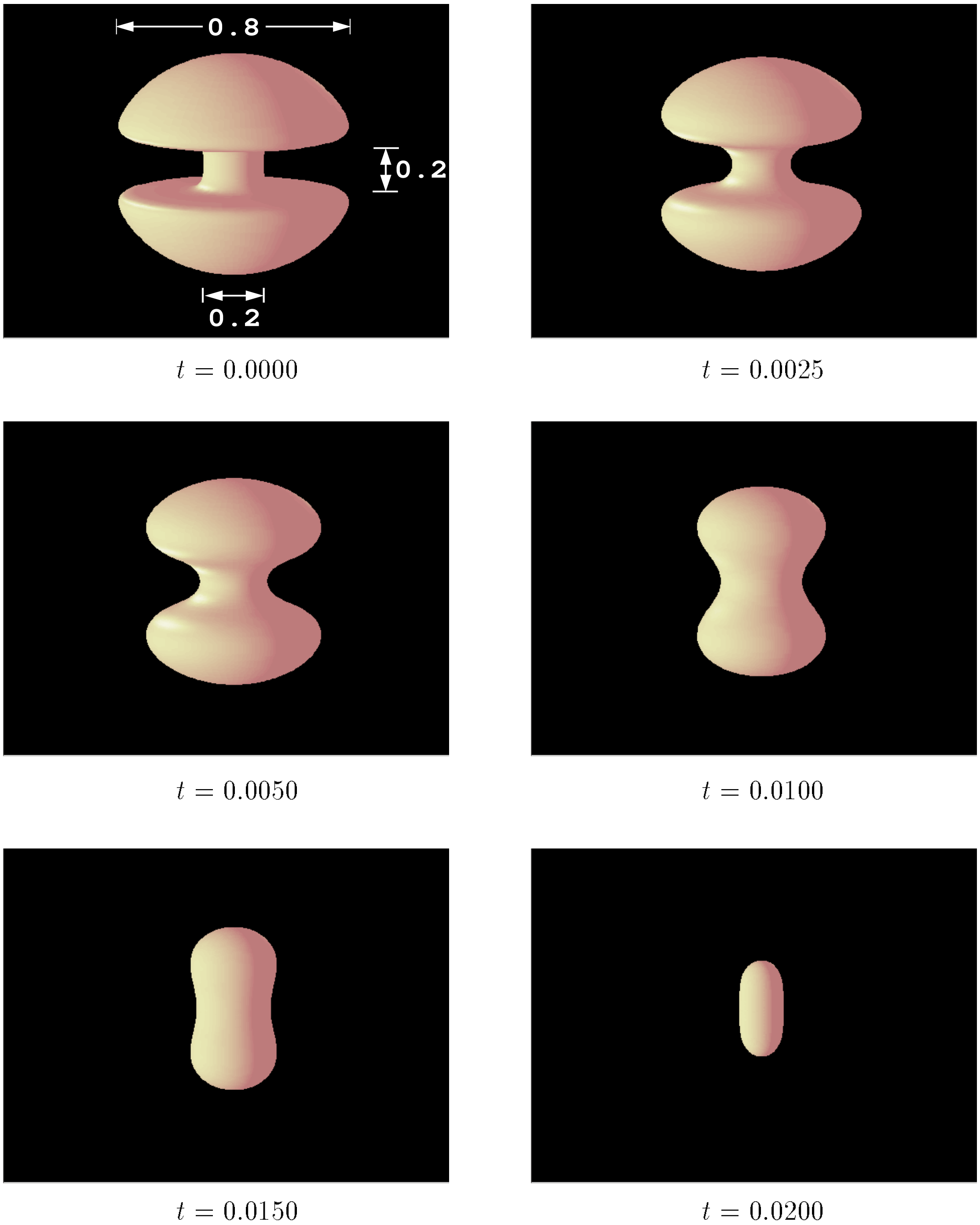


Figure 5.6: Thick-Stemmed Barbell Moving by Mean Curvature Motion

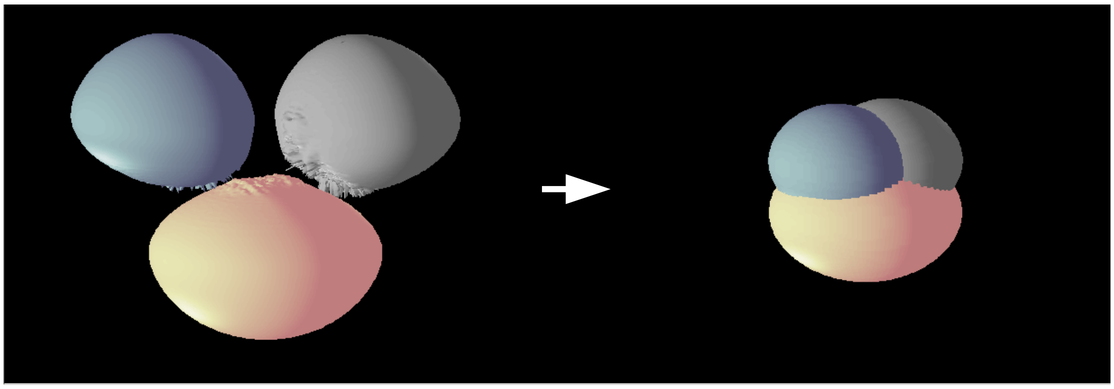


Figure 5.7: Composition of a Junction

This method draws each of the visible surfaces between the opaque and clear regions. It also constructs several artificial surfaces which are not displayed by *surfl()* since they are hidden. See, for example, Figure 5.7. It is an interesting fact that these artificial surfaces are often nonsmooth since the corresponding values of U_j and U_{clear} are zero to within the tolerance of the line search algorithm used in Visualization I.

Visualization II has been applied to a variety of surfaces with junctions. Indeed, each of the movie frames displayed in the next section was constructed using this approach.

5.2.2 Numerical Experiments

We now report on experiments for the motion by mean curvature of surfaces with junctions. Throughout this section, a trivial treatment of the finest subregions was used (see Section 3.3.1) with a finest cell width of $h = \frac{1}{128}$.

Three Phase Example

From Section 5.1.2, we saw that the MBO-method naturally treats two-phase problems in three dimensions. Multiple phase problems have also been evolved using the method.

For example, Figure 5.8 displays the motion by mean curvature of a cylindrical, three phase shape using a step size $\tau = 0.0005$. From these images, we see that the central blue phase pinches off to form two spherical regions, which eventually disappear.

Multiple Phase Example

The previous example considered the motion by mean curvature of three phases meeting at 120-120-120 degree angles. The evolution of four-phase junctions may also be studied using our new, spectral discretization of the MBO-method. For example, Figure 5.9 displays the motion of a spherical four-phase shape using a step size, $\tau = 0.0004$. From these images, we see that the four-phase junction is stable under mean curvature motion, as is expected from experimental studies of recrystallized metal [38].

5.3 A Nonlocal Model

The bulk of this thesis has been concerned with the motion by mean curvature of interfaces. We now consider an extension of this motion to a nonlocal model.

Consider a collection of disjoint interfaces, Γ_i, which separate a number of regions. If we evolve these interfaces using a normal velocity

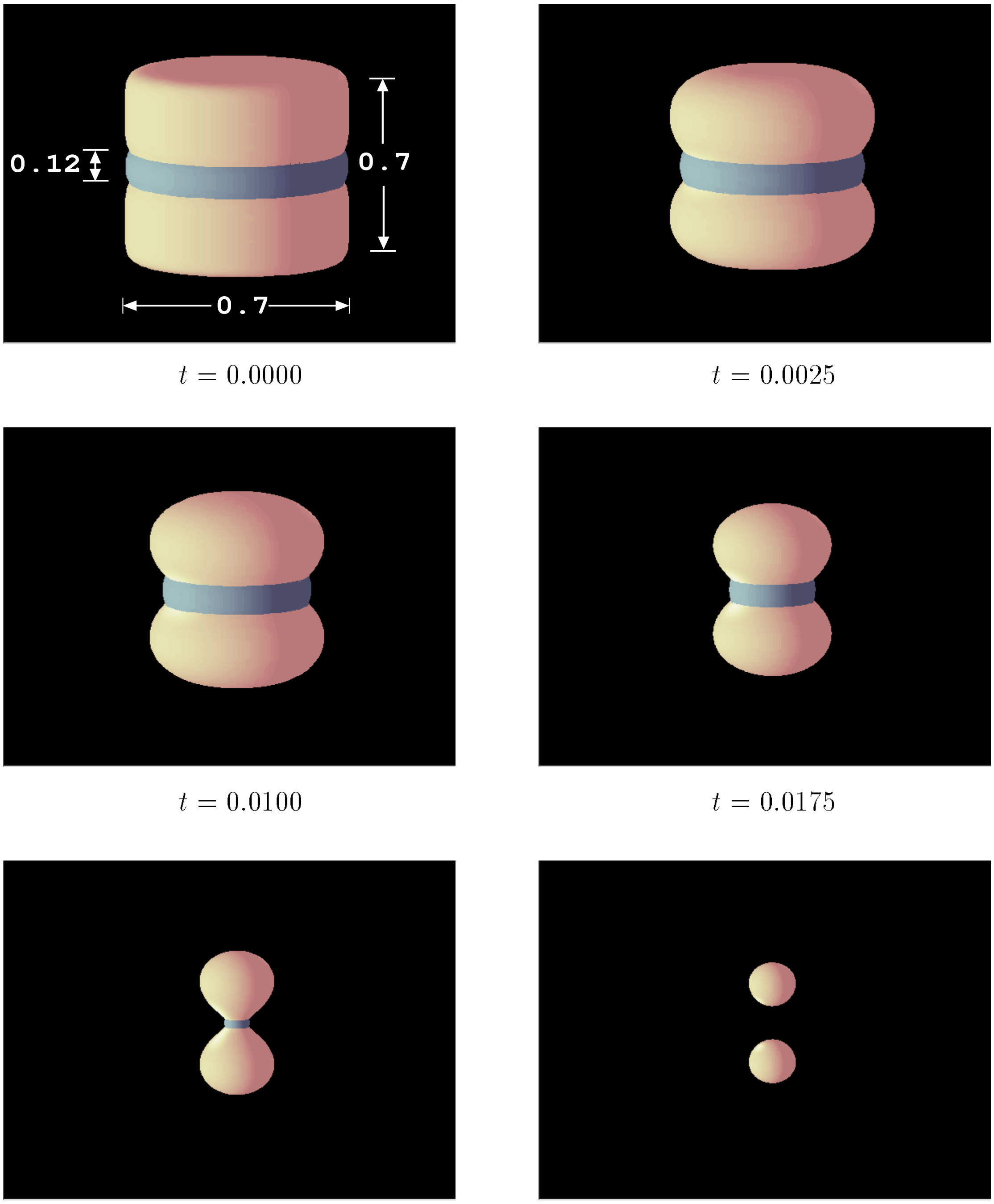
$$v_n(\vec{x}, t) = \kappa(\vec{x}, t) - \kappa_a(t) \tag{5.34}$$

where $\kappa(\vec{x}, t)$ and $\kappa_a(t)$ are the mean curvature and average mean curvature[1] respectively, then we obtain a nonlocal volume-preserving flow [57]. This motion is of interest

[1] Here, the average mean curvature is given by

$$\kappa_a = \left(\frac{1}{\sum_i |\Gamma_i|} \right) \left(\sum_i \int_{\Gamma_i} \kappa_i \right)$$

where $|\Gamma_i|$ is the perimeter of the interface of Γ_i.

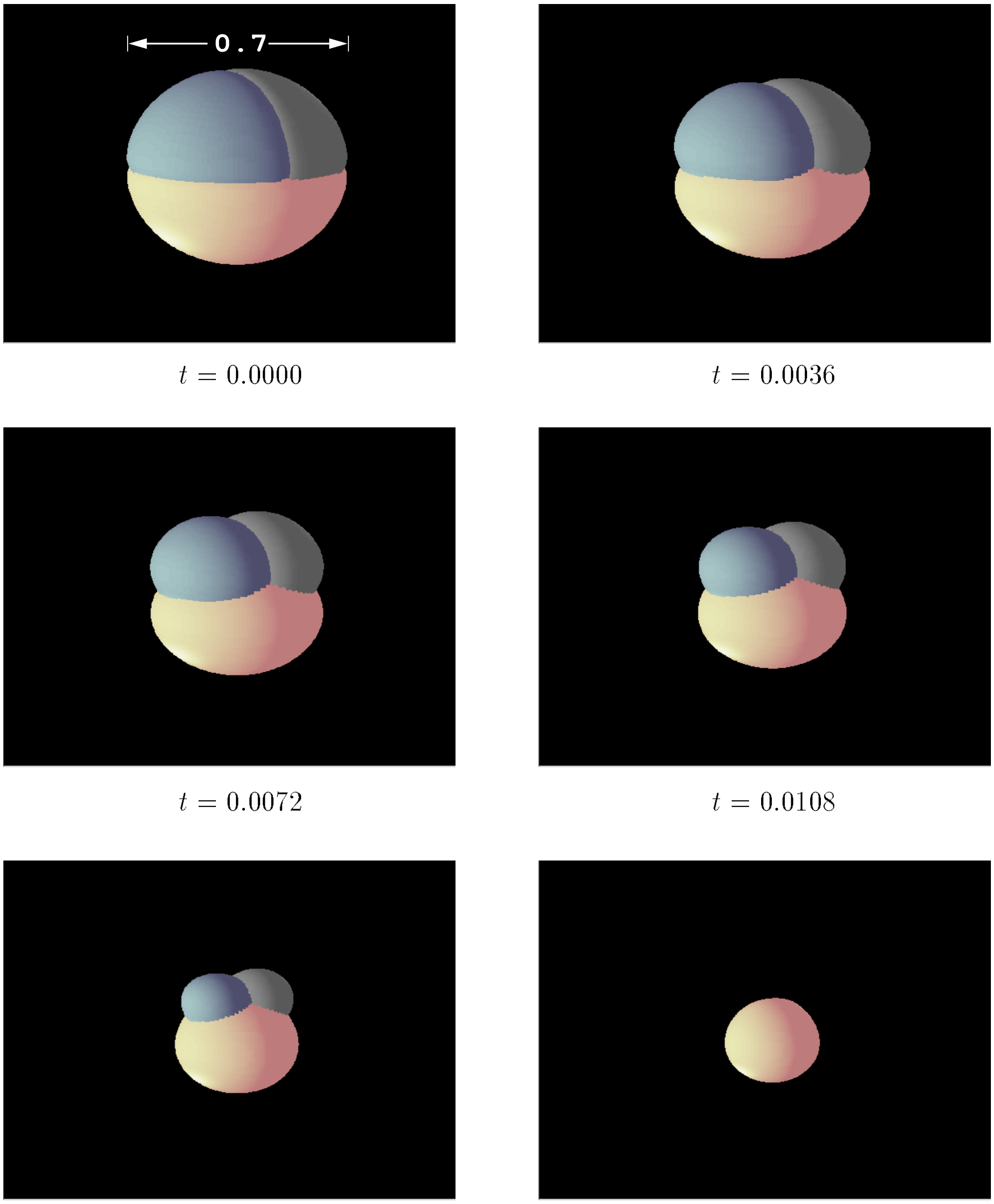


$t = 0.0000$ $t = 0.0025$

$t = 0.0100$ $t = 0.0175$

$t = 0.0220$ $t = 0.0260$

Figure 5.8: Three Phase Example Moving by Mean Curvature Motion



$t = 0.0000$ $t = 0.0036$

$t = 0.0072$ $t = 0.0108$

$t = 0.0144$ $t = 0.0180$

Figure 5.9: Four Phase Example Moving by Mean Curvature Motion

because it arises as a nonlocal model for binary alloys [57] and because it gives a possible smoothing for generating multiscale representations of planar curves [64].

We now develop a simple modification of the MBO-method to treat this nonlocal model in two dimensions. An extension to the three dimensional case is straightforward.

We begin by noting that an approximation to the desired motion (5.34) is obtained at each step if we track the level set

$$\frac{1}{2} - \frac{1}{2}\kappa_a(t)\sqrt{\frac{\tau}{\pi}} \tag{5.35}$$

rather than the usual level set of one-half [45]. Thus, phase areas remain approximately constant provided we follow the contour (5.35). Since U is continuous at any time t before sharpening, the area enclosed by a level set c,

$$A(c,t) = \text{Area}(\{\vec{x} : U(\vec{x},t) \leq c\})$$

is a strictly increasing and hence invertible function of c. Thus, the desired contour (5.35) can be approximated by the level set that preserves phase areas. Applying this result to the MBO-method gives a simple algorithm for computing solutions to the nonlocal model:

Nonlocal Curvature Algorithm

To obtain an approximation to the nonlocal curvature model (5.34), we may carry out the MBO-method for two phases given in Section 2.1 using the following as a replacement for step (3):

(3a) Find the level set that preserves phase areas; i.e., determine the value c satisfying

$$A(c,t) = A\left(\frac{1}{2}, 0\right). \tag{5.36}$$

Solving (5.36) for c may be accomplished by a variety of line search algorithms. For example, [11] gives an efficient and reliable approach based on a combination

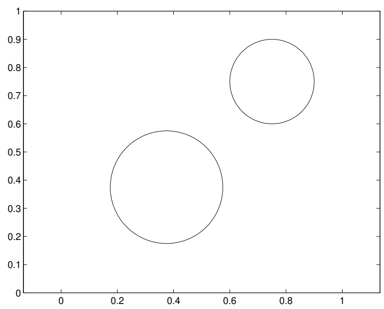


Figure 5.10: A Test Problem for the Nonlocal Curvature Algorithm

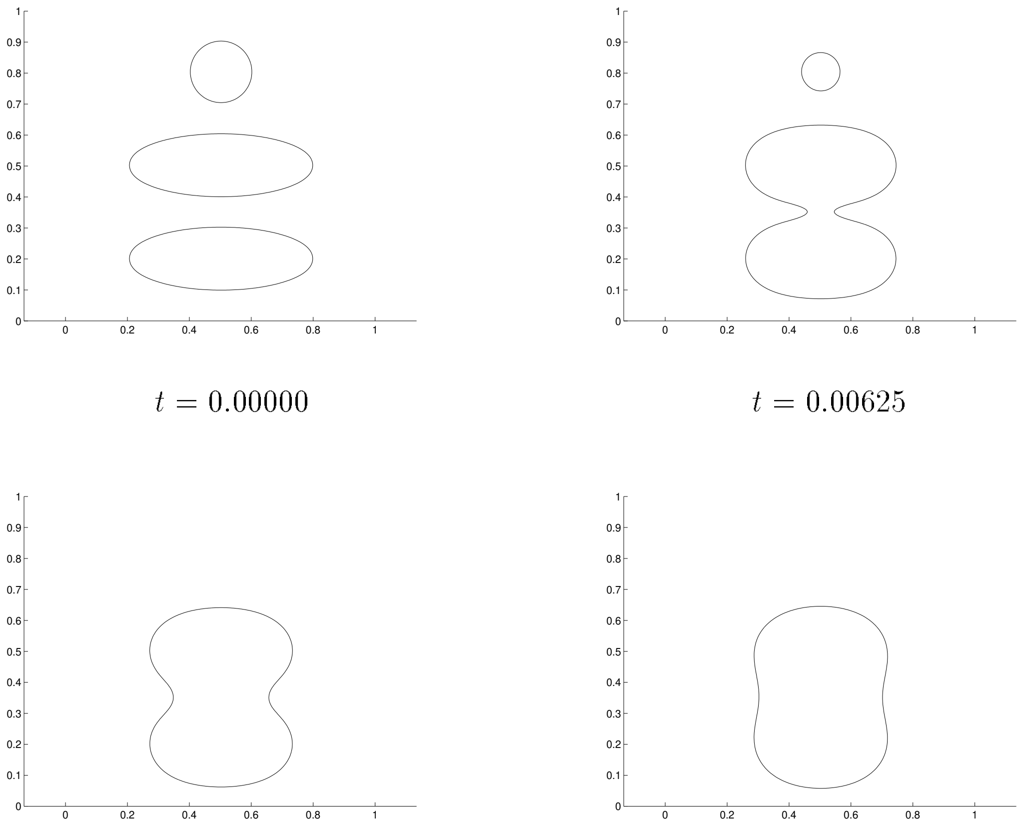
of bisection, secant and inverse quadratic interpolation methods. Note that this step is relatively simple to implement since the area of the phase we are following is given by the Fourier coefficient $c_{00}(t)$.

(3b) Carry out step (3) of the MBO-method using the contour c rather than the usual choice of one-half.

To test how well this method approximates the nonlocal model, we consider the motion of two circles with initial radii 0.2 and 0.15 (see Figure 5.10). Using the nonlocal curvature algorithm, the area of the smaller circle after a time $t = 0.02$ was compared to the exact answer, 0.044508, which was found by integration of (5.34). The results for a number of experiments are given in Table 12, below.

τ	*Error*	*Conv. Rate*
0.004	-0.0105	-
0.002	-0.00389	1.43
0.001	-0.00182	1.10
0.0005	-0.00104	0.80

Table 12. Results for the Nonlocal Curvature Algorithm



$t = 0.00000$ $t = 0.00625$

$t = 0.00875$ $t = 0.01250$

Figure 5.11: Nonlocal Model Which Preserves Area

These results indicate that the error decreases with τ. Unfortunately, our tests did not obtain a clear convergence rate.

Examples involving topological changes can be naturally handled by the method. For example, Figure 5.11 displays the motion of three regions using a step size of $\tau = 0.00025$ and a finest cell width of $h = \frac{1}{512}$. From these images, it is clear that the large elliptical regions grow at the expense of the smaller circle. Before long, the two elliptical regions merge and the circle disappears. As we expect, this final shape slowly smoothes to become more circular.

Chapter 6

Conclusions

6.1 Summary

In this thesis, we have considered a recent method [47, 48] for the motion by mean curvature of curves and surfaces. This method (MBO) naturally handles complicated topological changes with junctions. Because it produces approximations to these problems more efficiently than other methods, it is often the only practical choice for the accurate treatment of three dimensional surfaces involving junctions.

The usual finite difference discretization of the MBO-method has a number of limitations. For example, a very fine mesh must be used (see restriction (2.7)) to prevent the front from becoming stationary. Satisfying such a restriction globally is often computationally impractical when accurate results are needed. Furthermore, each step produces an error in the position of the front which is comparable in size to the mesh spacing. To preserve the overall accuracy of the method for smooth curves, this leads to a work estimate of $\mathcal{O}\left(\frac{1}{\tau^3}\right)$ operations per step for an optimized finite difference approach. (Here, $\tau \ll 1$ represents the time-length of each step of the MBO-method.) The usual finite difference approach also produces irregular errors, which precludes the use of extrapolation in τ for obtaining a higher order accuracy.

To overcome these, and other, limitations we have proposed a new, spectral discretization of the MBO-method which we have found is often more than 1000 times faster than the usual finite difference approach. This method uses a Fourier cosine tensor product to discretize the heat equation which arises at each τ-step of the MBO-method. The corresponding Fourier coefficients are determined using a quadrature with a piecewise linear approximation to the surface. This approach leads to several advantages over the usual finite difference discretization especially when used in combination with the recent, unequally spaced fast Fourier transform method given in [5].

Firstly, local refinement is much easier to carry out within a quadrature, rather than within a discretization of a differential equation (cf. [49]). Indeed, our proposed method recursively refines near the front to essentially eliminate the restriction (2.7), which plagues finite difference discretizations of the method. Furthermore, the piecewise linear approximation of the front which we use is more accurate than the crude approximation which arises for the finite difference approach. Indeed, we show that our new, spectral method requires only $\mathcal{O}(\frac{1}{\tau}\log^2(\tau))$ operations per step to preserve the overall accuracy of the MBO-method for smooth curves. This compares very favourably to the $\mathcal{O}\left(\frac{1}{\tau^3}\right)$ operation count that arises for an idealized finite difference approach. In practice, however, even this bound seems optimistic since no finite difference code exists (to the best of our knowledge) which requires fewer than $\mathcal{O}\left(\frac{1}{\tau^4}\right)$ operations per step.

Our proposed method also has the advantage that it allows for higher order extrapolations in τ since it essentially eliminates irregular spatial errors. By essentially eliminating these spatial errors we also produce an algorithm which is nearly Euclidean invariant and hence more attractive for certain image enhancement applications (e.g., [42]).

Further improvements in efficiency are obtained for the new method by neglecting high order Fourier modes which correspond to rapidly decaying solution transients. Gains also arise by carrying out the time integration exactly, rather than by a time-stepping method.

New analytic and experimental results are also given to explain important properties of the MBO-method, such as the approximation error. In particular, an asymptotic expansion for the position of the front is derived to show that the method is first order in τ for smooth curves. For nonsmooth corners and singularities, numerical experiments are carried out to demonstrate that an order $\tau \log(\tau)$ error arises. Higher order results for two-phase problems are obtained using extrapolation in the Fourier coefficients

An asymptotic expansion for the angles of a two dimensional triple junction is also derived. This expansion indicates that each step of the MBO-method produces an $\mathcal{O}(\sqrt{\tau})$ error in the junction angles which rapidly dissipates during subsequent steps. This result is of interest because it explains the stability of junction angles and suggests that an $\mathcal{O}(\sqrt{\tau})$ error arises for problems with junctions. Numerical experiments confirm that an $\mathcal{O}(\sqrt{\tau})$ error occurs, and indicate that extrapolation can be used in conjunction with the proposed method to produce a marked improvement in the error.

Finally, the improved utility of the new method is demonstrated by approximating and visualizing the motion by mean curvature of a number of three dimensional surfaces. Specifically, we give examples of two-phase, barbell-shaped regions which can undergo topological breaking depending on the width of the initial stem. Multiple phase motions are also approximated for examples involving three and four-phase junctions. We conclude our results with a simple extension of the MBO-method to a nonlocal curvature model that preserves phase volumes.

6.2 Future Research Directions

There are many directions for future work in diffusion-generated motion by mean curvature.

A detailed theoretical investigation of the method would be desirable. In particular, a convergence proof for the case of junctions would be of great interest. Unfortunately, the mathematical tools which have been applied to produce rigorous convergence proofs for two-phase problems with topological changes cannot be readily extended to junctions (see, e.g., [25]). However, because no method has been proven to converge for junctions we feel that even a formal convergence proof away from singularities would be an important achievement.

Application of the MBO-method to different curvature-dependent motions would also be of interest. Currently, arbitrary junction angles can be treated by the method [45]. Affine velocity motions for two-phase problems can also be carried out [45]. Extending the method to affine velocity problems with nonsymmetric junctions (cf. [74]) would be useful for modelling a combination of surface and bulk effects in idealized grain growth applications [71]. Extensions to anisotropic mean curvature flows [71], complicated domain geometries [20] and mixed boundary conditions [20] would also be of interest.

Experimental studies of mean curvature motion for surfaces with junctions might also be carried out using our realization method for the MBO-method. For example, self-similar solutions (cf. [21]) and singularities (cf. [22]) arising from mean curvature flow might be studied. Further studies of nonsmooth corners might also be undertaken by comparing the results of the MBO-method with known similarity solutions [65].

Certainly, effective visualizations of two-phase surfaces have been carried out [21]. Methods for the improved visualization of junctions should also be considered. We have found the effective display of these surfaces to be rather challenging since features of interest (e.g., moving junctions) can become obscured by nearby regions.

Finally, novel applications of junctions could also be investigated. In particular, the motion by mean curvature of junctions may be of value in certain image enhancement applications (cf. [19, 52, 23]).

Bibliography

[1] D. Adalsteinsson and J.A. Sethian. A fast level set method for propagating interfaces. *Journal of Computational Physics*, 118(2):269–277, 1995.

[2] L. Alvarez, P. Lions, and J. Morel. Image selective smoothing and edge detection by nonlinear diffusion: II. *SIAM J. Numerical Analysis*, 29(3):845–866, 1995.

[3] U.M. Ascher, S.J. Ruuth, and B.T.R. Wetton. Implicit-explicit methods for time-dependent PDE's. *SIAM J. Numerical Analysis*, 32(3):797–823, 1995.

[4] G. Barles and C. Georgelin. A simple proof of convergence for an approximation scheme for computing motions by mean curvature. *SIAM Journal of Numerical Analysis*, 32(2):484–500, 1995.

[5] G. Beylkin. On the fast Fourier transform of functions with singularities. *Applied and Computational Harmonic Analysis*, 2:363–381, 1995.

[6] J. Bloomenthal. Polygonization of implicit surfaces. *Computer Aided Geometric Design*, 5(4):341–355, November 1988.

[7] C. De Boor. *A Practical Guide to Splines*. Springer-Verlag, 1978.

[8] J.P. Boyd. *Chebyshev & Fourier Spectral Methods*. Springer-Verlag, 1989.

[9] K.A. Brakke. The surface evolver. *Experimental Mathematics*, 1(2):141–165, 1992.

[10] A. Brandt. Multilevel computations of integral transforms and particle interactions with oscillatory kernels. *Computer Physics Communications*, 65:24–38, 1991.

[11] R. Brent. *Algorithms for Minimization Without Derivatives*. Prentice-Hall, 1973.

[12] L. Bronsard and Kohn. Motion by mean curvature as the singular limit of Ginzburg-Landau dynamics. *Journal of Differential Equations*, 90(2):211–237, 1991.

[13] L. Bronsard and F. Reitich. On three-phase boundary motion and the singular limit of a vector-valued Ginzburg-Landau equation. *Arch. Rat. Mech.*, 124:355–379, 1993.

[14] L. Bronsard and B.T.R. Wetton. A numerical method for tracking curve networks moving with curvature motion. *Journal of Computational Physics*, 120(1):66–87, 1993.

[15] R.L. Burden and J.D. Faires. *Numerical Analysis*. PWS-Kent Pub. Co., 1993.

[16] G. Caginalp. The dynamics of a conserved phase field system: Stefan-like, Hele-Shaw, and Cahn-Hilliard models as asymptotic limits. *IMA J. applied Math.*, 44(1):77–94, 1990.

[17] C. Canuto, M.Y. Hussaini, A. Quarteroni, and T.A. Zang. *Spectral Methods in Fluid Dynamics*. Springer-Verlag, 1987.

[18] B.W. Char, K.O. Geddes, G.H. Gonnet, B.L. Leong, M.B. Monagan, and S.M. Watt. *Maple V Language Reference Manual*. Springer-Verlag, 1991.

[19] Y. Chen. Segmentation and association among lines and junctions for a line image. *Pattern Recognition*, 27(9):1135–1157, 1994.

[20] D.L. Chopp. Computing minimal surfaces via level set curvature flow. *Journal of Computational Physics*, 106(1):77–91, 1993.

[21] D.L. Chopp. Computation of self-similar solutions for mean curvature flow. *Experimental Mathematics*, 3(1):1–15, 1994.

[22] D.L. Chopp and J.A. Sethian. Flow under mean curvature: singularity formation, minimal surfaces and geodesics. *Experimental Mathematics*, 2(4):235–255, 1992.

[23] M. Demi. Contour tracking by enhancing corners and junctions. *Computer Vision and Image Understanding*, 63(1):118–134, 1996.

[24] A. Dutt and V. Rokhlin. Fast Fourier transform for nonequispaced data. *SIAM Journal on Scientific and Statistical Computing*, 14(6):1368–1393, 1993.

[25] L.C. Evans. Convergence of an algorithm for mean curvature motion. *Indiana University Mathematics Journal*, 42:553–557, 1993.

[26] L.C. Evans, H.M. Soner, and P.E. Souganidis. Phase transitions and generalized motion by mean curvature. *Communications on Pure and Applied Mathematics*, 45(9):1097–1123, 1992.

[27] L.C. Evans and J. Spruck. Motion of level sets by mean curvature, I. *J. Differential Geometry*, 23:69–96, 1986.

[28] R. E. Ewing. Adaptive mesh refinements in large-scale fluid flow simulation. In *International Conference on Accuracy Estimates and Adaptive Refinements in Finite Element Computations*, pages 299–314, Lisbon, Portugal, 1984.

[29] V.E. Fradkov, M.E. Glicksman, M. Palmera, J. Nordberg, and K. Rajan. Topological rearrangements during 2D normal grain growth. *Physica D*, 66:50–60, 1993.

[30] H. Frost, C. Thompson, C. Howe, and J. Whang. A two-dimensional computer simulation of capillarity-driven grain growth: preliminary results. *Scripta Metallurgica*, 22:65–70, 1988.

[31] M. Gage and R.S. Hamilton. The heat equation shrinking convex plane curves. *J. Differential Geometry*, 23:69–96, 1986.

[32] C.W. Gear. *Numerical initial-value problems in ordinary differential equations*. Prentice-Hall, 1971.

[33] J. Glazier, M. Anderson, and G. Grest. Coarsening in the two-dimensional soap froth and the large-Q Potts model: a detailed comparison. *Philosophical Magazine B*, 62:615–645, 1990.

[34] W.B. Gragg. On extrapolation algorithms for ordinary initial-value problems. *SIAM Journal on Numerical Analysis*, 2:384–403, 1965.

[35] M.A. Grayson. A short note on the evolution of a surfaces by its mean curvature. *Duke Mathematical Journal*, 58(3):555–558, 1989.

[36] M.A. Grayson. Shortening embedded curves. *Annals of Mathematics*, 129:71–111, 1989.

[37] E. Hairer, S.P. Norsett, and G. Wanner. *Solving Ordinary Differential Equations I*. Springer-Verlag, 1987.

[38] D. Harker and E.R. Parker. Grain shape and grain growth. In *Transactions of the A.S.M*, pages 156–161, Cleveland, 1945.

[39] G. Huisken. Flow by mean curvature of convex surfaces in spheres. *J. Differential Geometry*, 20:237–266, 1984.

[40] T. Ilmanen. Convergence of the Allen-Cahn equation to Brakke's motion by mean curvature. *Journal of Differential Geometry*, 38(2):417–461, 1993.

[41] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1987.

[42] B.B. Kimia, A.R. Tannenbaum, and S.W. Zucker. Shapes, shocks, and deformations I: The components of two-dimensional shape and the reaction-diffusion space. *International Journal of Computer Vision*, 15(3):189–224, 1995.

[43] J.J. Koenderink and A.J. van Doorn. Dynamic shape. *Biological Cybernetics*, 53:383–396, 1986.

[44] W.E. Lorensen and H.E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In M.C. Stone, editor, *Computer Graphics (SIGGRAPH '87 Proceedings)*, volume 21(4), pages 163–169, Anaheim, California, July 1987.

[45] P. Mascarenhas. Diffusion generated by mean curvature. CAM Report 92-33, University of California, Los Angeles, 1992.

[46] The MathWorks. *MATLAB. High-Performance Numeric Computation and Visualization Software*. Springer-Verlag, 1992.

[47] B. Merriman, J. Bence, and S. Osher. Diffusion generated motion by mean curvature. In J.E. Taylor, editor, *Computational Crystal Growers Workshop*, pages 73–83. American Mathematical Society, Providence, Rhode Island, 1992.

[48] B. Merriman, J. Bence, and S. Osher. Motion of multiple junctions: a level set approach. *Journal of Computational Physics*, 112(2):334–363, 1994.

[49] B. Milne. *Adaptive Level Set Methods Interfaces*. PhD thesis, University of California, Berkeley, 1995.

[50] F. Mokhtarian and A.K. Mackworth. A theory of multiscale, curvature-based shape representation for planar curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(8):789–805, 1992.

[51] W.W. Mullins. Two-dimensional motion of idealized grain boundaries. *Journal of Applied Physics*, 27(8):900–904, 1956.

[52] J.A. Noble. Finding half boundaries and junctions in images. *Images and Vision Computing*, 10(4):219–232, 1992.

[53] R.H. Nochetto, M. Paolini, and C. Verdi. A dynamic mesh algorithm for curvature dependent evolving interfaces. *Journal of Computational Physics*, 123(2):296–310, 1996.

[54] S. Osher and J.A. Sethian. Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics*, 79:12–49, 1988.

[55] W.H. Press and G.B. Rybicki. Fast algorithm for spectral analysis of unevenly sampled data. *Astrophysical Journal*, 338:227–280, 1989.

[56] F. Reitich and H.M. Soner. Three-phase boundary motions under constant velocities. I: The vanishing surface tension limit. Technical Report 94-NA-015, Centre for Nonlinear Analysis, Carnegie Mellon University, 1994.

[57] J. Rubinstein and P. Sternberg. Nonlocal reaction-diffusion equations and nucleation. *IMA J. Appl. Math.*, 48:248–264, 1992.

[58] J. Rubinstein, P. Sternberg, and J.B. Keller. Fast reaction, slow diffusion and curve shortening. *SIAM J. Appl. Math.*, 49(1):116–133, 1989.

[59] J. Rubinstein, P. Sternberg, and J.B. Keller. Reaction-diffusion processes and evolution to harmonic maps. *SIAM J. Appl. Math.*, 49(6):1722–1733, 1989.

[60] S.J. Ruuth. An algorithm for generating motion by mean curvature. In *Proc. 12th International Conference on Analysis and Optimization of Systems Images, Wavelets and PDE's*, Paris, France, 1996.

[61] H. Samet. *Applications of spatial data structures: computer graphics, image processing, and GIS*. Addison-Wesley, 1990.

[62] H. Samet. *The design and analysis of spatial data structures*. Addison-Wesley, 1990.

[63] G. Sapiro and A. Tannenbaum. Affine invariant scale-space. *International Journal of Computer Vision*, 11(1):25–44, 1993.

[64] G. Sapiro and A. Tannenbaum. Area and length preserving geometric invariant scale-space. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(1):67–72, 1995.

[65] D.W. Schwendeman. A front dynamics approach to flow by mean curvature. Technical report, Department of Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, New York, 12180-3590, 1996.

[66] J.A. Sethian. Curvature flow and entropy conditions applied to grid generation. *Journal of Computational Physics*, 115:440–454, 1994.

[67] J.A. Sethian. Theory, algorithms, and applications of level set methods for propagating interfaces. *Acta Numerica*, 5:309–395, 1996.

[68] E. Sorets. Fast Fourier transforms of piecewise constant functions. *Journal of Computational Physics*, 116:369–379, 1995.

[69] J.C. Strikwerda. *Finite Difference Schemes and Partial Differential Equations*. Wadsworth & Brooks/Cole, 1989.

[70] T.D. Sullivan. A technique for convolving unequally spaced samples using fast Fourier transforms. Technical report, Sandia Report, 1990.

[71] J.E. Taylor, J.W. Cahn, and C.A. Handwerker. I-Geometric models of crystal growth. *Acta Metall. Meter.*, 40(7):1443–1474, 1992.

[72] D. Terzopoulos, A. Witkin, and M. Kass. Constraints on deformable models: Recovering 3D shape and nonrigid motion. *Artificial Intelligence*, 36(1):91–123, 1988.

[73] J.J. Tyson and J.P. Keener. Singular perturbation theory of traveling waves in excitable media. *Physica D*, 32:327–361, 1988.

[74] H. Zhao, T. Chan, B. Merriman, and S. Osher. A variational level set approach to multiphase motion. CAM Report 95-36, University of California, Los Angeles, 1995.

Appendix A

An Estimate for the Number of Basis Functions

We now estimate how many basis functions are required to accurately represent the position of a smooth, two dimensional closed curve (e.g., Figure 1.1 or Figure 2.1) after a time, τ. Specifically, we compare the position of the level set $\frac{1}{2}$ for

$$U_\tau^n(x,y) = \sum_{i,j=0}^{n-1} c_{ij} \exp(-\pi^2[i^2+j^2]\tau) \cos(\pi i x) \cos(\pi j y)$$

and

$$U_\tau^\infty(x,y) = \sum_{i,j=0}^{\infty} c_{ij} \exp(-\pi^2[i^2+j^2]\tau) \cos(\pi i x) \cos(\pi j y)$$

to find a bound, n_c, such that for all $n \geq n_c$ the approximate interface locations for U_τ^n and U_τ^∞ differ by at most $\epsilon + \mathcal{O}(\epsilon\tau)$.

There are three steps in the derivation:

1. Lemma A.1 demonstrates that if

   $$|U_\tau^\infty - U_\tau^n|_\infty \leq \frac{\epsilon}{2\sqrt{\pi\tau}}$$

   then the approximate interface locations for U_τ^n and U_τ^∞ differ by at most

   $$\epsilon + \mathcal{O}(\epsilon\tau).$$

2. Lemma A.2 derives an estimate of the size of the Fourier coefficients, c_{ij}.

3. Finally, Theorem A.1 estimates the sum of the neglected Fourier terms of U_τ^n and shows that a safe choice of n is given by

   $$n \geq n_c = \sqrt{\frac{|\ln(\frac{\epsilon}{16\pi L})|}{\pi^2\tau}}$$

where L is the length of the curve at the beginning of the current step.

Lemma A.1 *If the initial interface is smooth (four times continuously differentiable) and*

$$|U_\tau^\infty - U_\tau^n|_\infty \leq \frac{\epsilon}{2\sqrt{\pi\tau}},$$

then the $\frac{1}{2}$ *level sets for* U_τ^∞ *and* U_τ^n *are at most a distance*

$$\epsilon + \mathcal{O}(\epsilon\tau)$$

apart.

Proof. Begin by orienting the shape according to Figure 4.1 for some arbitrary point on the initial interface. Following the notation of Section 4.1.1, we see from Equation (4.28) that after a time τ the level set $(\frac{1}{2} + \frac{\epsilon}{2\sqrt{\pi\tau}})$ is a distance

$$\tilde{z}(\tau) \quad = \quad g^{(2)}\tau - \epsilon + \left[\frac{1}{2}g^{(4)} - \frac{2}{3}(g^{(2)})^3\right]\tau^2 + \mathcal{O}(\epsilon\tau) + \mathcal{O}(\tau^3)$$

above the initial interface position[1]. Comparing to the result for the level set $\frac{1}{2}$,

$$z(\tau) \quad = \quad g^{(2)}\tau + \left[\frac{1}{2}g^{(4)} - \frac{2}{3}(g^{(2)})^3\right]\tau^2 + \mathcal{O}(\tau^3),$$

and noting that all terms not involving a factor of ϵ are identical, we see that

$$|\tilde{z}(\tau) - z(\tau)| \leq \epsilon + \mathcal{O}(\epsilon\tau). \quad \square$$

Lemma A.2 *If* $i \geq n$*, then the Fourier coefficients satisfy*

$$|c_{ij}| < \frac{4L}{n}$$

where L *is the length of the curve at the beginning of the current step.*

[1]We have assumed $\epsilon \leq \tau$. This will be true for any reasonable choice of ϵ since we expect to take $\mathcal{O}(\frac{1}{\tau})$ steps in total.

Proof. From Equation (3.12), we know that

$$
\begin{aligned}
|c_{ij}| &\leq 4\left|\iint\limits_{R_t} \cos(\pi i x)\cos(\pi j y)\; dA\right| \\
&\leq 4\left|\iint\limits_{R_t} \cos(\pi i x)\; dA\right|. \qquad \text{(A.37)}
\end{aligned}
$$

Over any rectangle, $\tilde{R} \subset R_t$, of width $\frac{2}{i}$

$$
\iint\limits_{\tilde{R}} \cos(\pi i x)\; dA = 0
$$

since $\int_{x_0}^{x_0+\frac{2}{i}} \cos(\pi i x)dx = 0$. Dividing the domain into rectangles of width $\frac{2}{i}$, we see that only subregions in contact with the interface contribute to the bound (A.37). See, for example, Figure A.1. Indeed, letting the height of each rectangle tend to zero, it is easy to see that only contributions within a horizontal distance $\frac{2}{i}$ from the interface contribute. Using this fact, and integrating only over half of each rectangle so the sign of $\cos(\pi i x)$ does not change,

$$
|c_{ij}| < \frac{4L}{i} \leq \frac{4L}{n}
$$

where L is the length of the interface at the beginning of the current step. □

We now give the main result of this section.

Theorem A.1 *Suppose that U_τ^n approximates U_τ^∞ and that the initial interface is smooth. If*

$$
n > n_c = \sqrt{\frac{|\ln(\frac{\epsilon}{16\pi L})|}{\pi^2 \tau}},
$$

then the interface approximations for U_τ^n and U_τ^∞ are within a distance

$$
\epsilon + \mathcal{O}(\epsilon\tau)
$$

of one another.

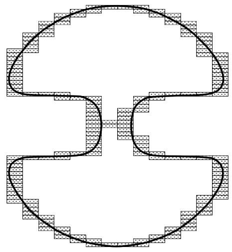


Figure A.1: Contributing Rectangles

Proof. We begin by estimating

$$
\begin{aligned}
|U_\tau^\infty - U_\tau^n|_\infty &= \Big| \sum_{\substack{i,j=0 \\ \max(i,j)\geq n}}^{\infty} c_{ij} \exp(-\pi^2[i^2+j^2]\tau)\cos(\pi i x)\cos(\pi j y)\Big|_\infty, \\
&\leq \sum_{\substack{i,j=0 \\ \max(i,j)\geq n}}^{\infty} |c_{ij}| \exp(-\pi^2[i^2+j^2]\tau), \\
&\leq \sum_{i=n}^{\infty}\sum_{j=0}^{\infty} (|c_{ij}|+|c_{ji}|)\exp(-\pi^2[i^2+j^2]\tau).
\end{aligned}
$$

Using Lemma A.2, this simplifies to give us

$$
\begin{aligned}
|U_\tau^\infty - U_\tau^n|_\infty &< \frac{8L}{n}\sum_{i=n}^{\infty}\sum_{j=0}^{\infty}\exp(-\pi^2[i^2+j^2]\tau), \\
&= \frac{8L}{n}\sum_{i=n}^{\infty}\exp(-\pi^2 i^2\tau)\sum_{j=0}^{\infty}\exp(-\pi^2 j^2\tau), \\
&< \frac{8L}{n}\left(\exp(-\pi^2 n^2\tau) + \int_n^\infty \exp(-\pi^2 n s\tau)\, ds\right)\left(1+\int_0^\infty \exp(-\pi^2 s^2\tau)\, ds\right) \\
&= \frac{8L}{n}\left(1+\frac{1}{\pi^{3/2}\sqrt{\tau}\sqrt{\pi n^2\tau}}\right)\left(1+\frac{1}{2}\frac{1}{\sqrt{\pi\tau}}\right)\exp(-\pi^2 n^2\tau)
\end{aligned}
$$

Now, assuming that $\pi n^2\tau \geq 1$ and $\tau \leq 1$ (this is consistent with the final result for any reasonable error tolerance), it is easy to show that

$$
|U_\tau^\infty - U_\tau^n|_\infty < \frac{8\sqrt{\pi}L}{\sqrt{\tau}}\exp(-\pi^2 n^2\tau).
$$

By Lemma A.1, we want

$$|U_\tau^\infty - U_\tau^n|_\infty \quad \leq \quad \frac{\epsilon}{2\sqrt{\pi\tau}},$$

so we seek an n such that

$$\frac{8\sqrt{\pi}L}{\sqrt{\tau}} \exp(-\pi^2 n^2 \tau) \quad \leq \quad \frac{\epsilon}{2\sqrt{\pi\tau}}.$$

Solving for n, we arrive at a bound,

$$n \geq n_c = \sqrt{\frac{|\ln(\frac{\epsilon}{16\pi L})|}{\pi^2\tau}}. \qquad \square$$