JOURNAL OF COMPUTATIONAL PHYSICS **144,** 603–625 (1998)
ARTICLE NO. CP986025

Efficient Algorithms for Diffusion-Generated Motion by Mean Curvature

Steven J. Ruuth

University of California, Department of Mathematics, 405 Hilgard Avenue, Los Angeles, California 90095-1555
E-mail: ruuth@math.ucla.edu

Received August 12, 1997; revised April 15, 1998

The problem of simulating the motion of evolving surfaces with junctions according to some curvature-dependent speed arises in a number of applications. By alternately diffusing and sharpening characteristic functions for each region, a variety of motions have been obtained which allow for topological mergings and breakings and produce no overlapping regions or vacuums. However, the usual finite difference discretization of these methods is often excessively slow when accurate solutions are sought, even in two dimensions. We propose a new, spectral discretization of these diffusion-generated methods 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 contributions of rapidly decaying solution transients. Indeed, numerical studies demonstrate that the new algorithm is often more than 1000 times faster than the usual finite difference discretization. Our findings are demonstrated on several examples.

Key Words: diffusion; interface; motion by mean curvature; multiple junctions; spectral method.

1. INTRODUCTION

In a variety of applications, one wants to follow the motion of a front that moves with some curvature-dependent speed. Such motions can be particularly challenging to approximate when more than two phase regions are present because junctions of moving surfaces can occur. To simulate the evolution of such models, a number of numerical methods have been developed.

Front tracking methods (see, e.g., [10, 5] and references therein), are often well-suited 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.

0021-9991/98 $25.00

Because complicated topological changes are often possible, front tracking methods can be impractical to implement, especially in more than two dimensions.

Other approaches also have limitations. Monte Carlo methods for Potts models can introduce unwanted anisotropy into the motion, due to the spatial mesh [29], and are typically too slow to find accurate approximations of the model. Phase field methods are often inherently too expensive for practical computation [16] because they represent the interface as an internal layer and, thus, require an extremely fine mesh (at least locally) to resolve this layer.

To address these concerns for the case of pure mean curvature flow (i.e., each interface moves with a normal velocity equal to its mean curvature, κ), a method was proposed by Merriman, Bence, and Osher (MBO) which alternately diffuses and sharpens characteristic functions for regions [15, 16]. This method naturally handles complicated topological changes with junctions in several dimensions and has been rigorously proven to converge when two phase regions are present [7, 2]. Furthermore, in the two-phase case, generalizations based on this method and on the "threshold growth dynamics" of Gravner and Griffeath [9] produce a variety of motions which depend on the principal curvatures of the surface and the local normal direction [13, 14, 24]. These *convolution-based methods* are particularly interesting because they may be viewed as the continuum limit of certain cellular automata evolution rules [9, 14, 24]. In the multiphase case, a generalization appropriate for the approximation of grain growth models [1, 4, 20] has been developed and studied [23]. This diffusion-generated approach allows each interface, Γ_{ij}, to move with a normal velocity,

$$v_{ij} = \gamma_{ij}\kappa_{ij} + e_i - e_j, \tag{1}$$

where e_p represents the constant bulk energy for the pth phase region (see Fig. 1). Unfortunately, the usual finite difference discretization of the MBO-method and its generalizations is often exceedingly slow when accurate results are sought, even in two dimensions.

Other methods for approximating the motion of interfaces have also been developed. The level set method of Osher and Sethian [18], for example, computes arbitrary curvature-dependent surface motion, including topological changes. Standard numerical PDE methods apply to accurately and efficiently discretize the equations for motion [19]. Although well suited for a wide variety of two phase problems, the original level set method does not apply to the motion of junctions. An extension to this case was ultimately developed in [16]. This extension treats the triple point in a similar manner to the MBO-method described above

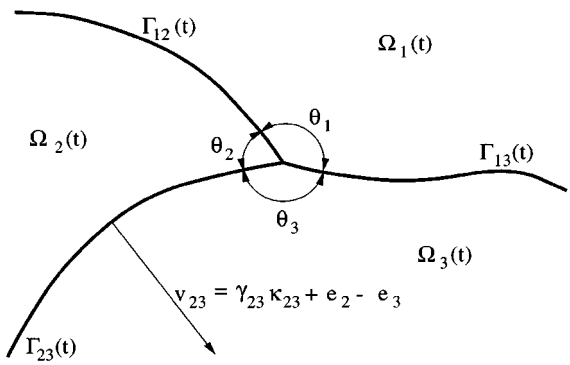

FIG. 1. The interfaces, Γ_{ij}, move with a normal velocity $v_{ij} = \gamma_{ij}\kappa_{ij} + e_i - e_j$ and are subject to angles $\theta_1, \theta_2, \theta_3$.

(which suggests a similar order of accuracy) but which has several additional difficulties. In particular, "vacuums" (parts of the domain that do not contain any phase region) form near junctions and a "redistancing" is required at each step of the algorithm [16]. This early method also "lacks (so far) a clear theoretical basis" [30] and remains largely unexplored, numerically.

More recently, a variational approach [30] based on the motion of level sets has been proposed to approximate the multiphase model (1). This interesting approach is especially well suited for treating problems with additional constraints. Unfortunately, it is unable to approximate many problems involving more than three phase regions. This is easily seen since only r independent γ_{ij} may be prescribed, where r is the number of phase regions. Furthermore, this method limits angles to the classical condition (see, e.g., [27])

$$\frac{\sin(\theta_1)}{\gamma_{23}} = \frac{\sin(\theta_2)}{\gamma_{13}} = \frac{\sin(\theta_3)}{\gamma_{12}}$$

at triple points.

In this paper, we propose a new, spectral discretization for the diffusion-generated methods which obtains greatly improved efficiency over the usual finite difference discretization. Although these algorithms are given (for simplicity) for the MBO-method, we note that they may also be applied to its generalizations described in [24, 14, 23]. Our algorithms, when applied to the MBO-method give a simple, fast way of approximating motion by mean curvature and, when applied to convolution-based methods, provide efficient computational schemes for the limiting motion of certain cellular automata rules or more general anisotropic curvature-dependent motions [24]. Furthermore, our algorithms when applied to the generalizations of [23] give a practical tool, not available hitherto, for accurately treating a wide variety of motions described by the multiphase model (1). An outline of the paper follows.

In Section 2, we give the MBO-method for two-phase and multiphase problems. For the case of the finite difference discretizations originally proposed [15], the selection of the step size is discussed and several limitations of the method are identified.

In Section 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.

Section 4 gives 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 Section 5, we use our algorithms to examine the numerical convergence properties of the MBO-method and apply our algorithms to the motion by mean curvature of surfaces. This section also demonstrates that extrapolation can be used in conjunction with our method to produce improved estimates of certain quantities of interest (e.g., phase areas).

2. THE MBO-METHOD

An algorithm for following interfaces propagating with a normal velocity equal to mean curvature was introduced by Merriman, Bence, and Osher [15, 16]. In this section, we describe the method for the two-phase and multiple-phase problems.

2.1. The Two-Phase Problem

Suppose we wish to follow an interface moving with a normal velocity equal to its mean curvature. The motion of such a surface may be approximated using the MBO-method for two regions:

MBO-METHOD (Two regions).

BEGIN

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

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

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

BEGIN

(2) Apply diffusion[1] to U for some time, Δt.

i.e., find $U(\mathbf{x}, j\Delta t)$ using $\begin{cases} U_t = \nabla^2 U, \\ \partial U/\partial n = 0 \quad \text{on } \partial\mathcal{D} \end{cases}$

starting from $U(\mathbf{x}, (j-1)\Delta t)$.

(3) "Sharpen" the diffused region by setting

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

END

END

For any time t, the level set $\{\mathbf{x} : U(\mathbf{x}, t) = \frac{1}{2}\}$ gives the location of the interface. Note that it is essential to apply the diffusion for only a short time, Δt, since only the initial motion generated by the diffusion is equal to motion by mean curvature (see [7, 2] for a precise statement).

2.2. Multiple Regions

To obtain a normal velocity equal to the mean curvature for symmetric junctions (e.g., a 120–120–120 degree junction in two dimensions), we may apply the MBO-method for multiple regions:

MBO-METHOD (Multiple (r) regions).

BEGIN

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

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

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

BEGIN

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

Apply diffusion to U_i for some time slice, Δt.

[1] Here we have selected zero flux boundary conditions to ensure that the curve meets the boundary at right angles, as is appropriate for certain grain growth models [5]. Alternatively, one may minimize the effects of the boundary by selecting nonreflecting boundary conditions, $\partial^2 U/\partial n^2 = 0$ (cf. [30]), or use Dirichlet conditions to produce a constrained motion.

$$\text{i.e., find } U_i(\mathbf{x}, j\Delta t) \text{ using } \begin{cases} \frac{\partial U_i}{\partial t} = \nabla^2 U_i, \\ \frac{\partial U_i}{\partial n} = 0 \quad \text{on } \partial\mathcal{D}. \end{cases}$$

(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.

END

END

For any time t, the interfaces are given naturally as the boundaries of the characteristic sets.

2.3. Time Step Selection

To accurately resolve the motion of features of the interface, it is also important to select Δt appropriately. Once a sufficiently small Δt is selected, a spatial discretization must be chosen that can resolve the motion of the interface. 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 [16]. This produces the restriction that

$$\text{grid spacing} \ll (\text{speed of motion of the interface}) \times \Delta t.$$

Letting κ be the curvature and Δx the grid spacing, we arrive at a restriction on Δx for the finite difference approach [16],

$$\Delta x \ll \kappa \Delta t. \tag{2}$$

As we shall see, the restriction (2) does not appear for the method that we propose in Section 3.

2.4. Limitations of Finite Difference Discretizations

Satisfying the restriction (2) derived in the previous section can be computationally impractical even for smooth, two-phase problems in two dimensions. Consider, for example, evolving the boundary of a spiral-shaped region according to a normal velocity equal to its curvature. Since the local curvature of the boundary of such a problem can vary tremendously, it may be impractical to satisfy (2) everywhere using a uniform mesh.

To achieve a more efficient finite difference algorithm, one might consider discretizing the MBO-method by placing a narrow band of grid points around the front. However, even this optimized finite difference approach can lead to a prohibitive number of operations at each step of the method.

For example, consider the motion by curvature of a smooth curve. For such a curve, the MBO-method is locally first order in Δt since each step of the method produces an $\mathcal{O}(\Delta t^2)$ error in the position of the front [22]. To preserve the overall accuracy of the method, grid points must be at most a distance $\mathcal{O}(\Delta t^2)$ apart since each step produces an error which is comparable to the mesh spacing. Noting that the front travels a distance $\mathcal{O}(\Delta t)$ per step of the method, it is clear that a minimum of $\mathcal{O}(1/\Delta t^3)$ grid points are needed to safely band a curve (see, e.g., Fig. 2). Thus, a minimum of $\mathcal{O}(1/\Delta t^3)$ 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

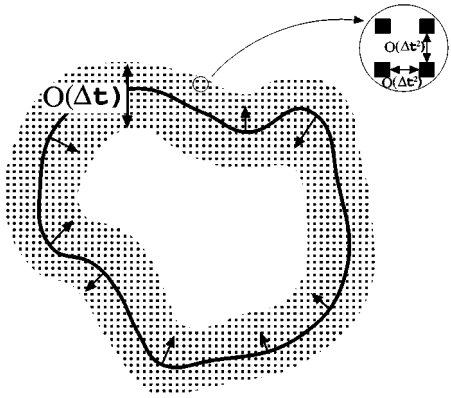

FIG. 2. A banded finite difference mesh.

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 3 and 4 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 when using finite differences. Our proposed method can essentially eliminate spatial errors by using a local refinement with a piecewise linear approximation of the front. In smooth, two-phase problems, the remaining error in the front position is of the form

$$c(\Delta t)^2 + \mathcal{O}(\Delta t)^3,$$

where c depends of the local geometry of the problem [22]. Thus, in two phase problems we expect that extrapolation in Δt can be used in conjunction with our proposed method to produce higher order accurate results. In the multiphase case the form of the error is not known, but numerical studies indicate that improved results are still obtained. See Section 5 and [21, 22].

To avoid the limitations of the original finite difference approach, a new, spectral method for realizing the MBO-method is introduced in the next section.

3. A NEW SPECTRAL METHOD

As we shall see later in this section, accurate computation of solutions using the usual finite difference discretization of the MBO-method can be expensive, even for simple

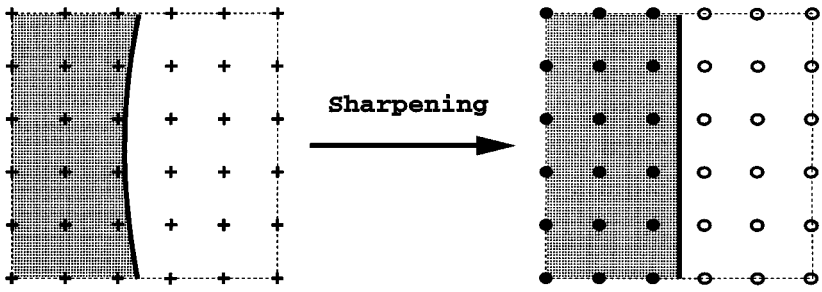

FIG. 3. Sharpening a shape.

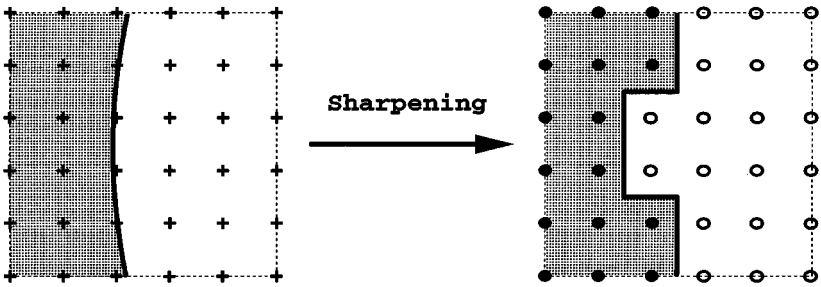

FIG. 4. Sharpening a perturbed shape.

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 section 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-phase 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

Carrying out diffusion-generated motion by mean curvature requires us to solve the heat equation

$$\begin{aligned} u_t &= \Delta u, \\ \frac{\partial u}{\partial n} &= 0 \quad \text{on } \partial\mathcal{D} \end{aligned} \tag{3}$$

repeatedly over time intervals of (possibly variable) length Δt, 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,

$$U(x, y, t) = \sum_{i,j=0}^{n-1} c_{ij} \exp(-\pi^2[i^2 + j^2][t - t_{\text{start}}]) \cos(\pi i x) \cos(\pi j y) \tag{4}$$

for $t_{\text{start}} \le t \le t_{\text{start}} + \Delta t$, where 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 Δt, however. After a sufficiently large time Δt, high frequency modes have dissipated. Since the problem is linear, different modes do not interact 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) at time Δt can be obtained using far fewer basis functions than might otherwise might be expected. Indeed, to approximate the position of the front to within a distance $\mathcal{O}(\epsilon)$, our implementations simply select an n satisfying

$$n \ge \sqrt{|\ln(\epsilon)|/\pi^2 \Delta t} \tag{5}$$

and verify the corresponding results by repeating the calculation with a different n (see also [22]).

3.2. Calculation of the Fourier Coefficients

The values of the Fourier coefficients, c_{ij}, of Eq. (4) must still be determined at the beginning of each time step (i.e., immediately following the sharpening of the previous step). In fact, we calculate these coefficients as part of the sharpening step using an adaptive quadrature method rather than a pseudospectral method. Begin by defining

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

to be the approximation of the phase we are following. By multiplying Eq. (4) at time $t = t_{\text{start}}$ by $\cos(\pi i x)\cos(\pi j y)$, integrating over the domain, and simplifying via the usual orthogonality conditions, we find

$$c_{ij} = \alpha_{ij} \int_0^1 \int_0^1 U(x, y, t_{\text{start}}) \cos(\pi i x) \cos(\pi j y)\, dx\, dy,$$

where

$$\alpha_{ij} = \begin{cases} 1, & \text{if } i = j = 0 \\ 4, & \text{if } i \neq 0;\ j \neq 0 \\ 2, & \text{otherwise.} \end{cases} \tag{6}$$

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

$$c_{ij} = \alpha_{ij} \iint_{R_t} \cos(\pi i x) \cos(\pi j y)\, dA. \tag{7}$$

Thus, simple functions must be integrated over a complicated, nonrectangular region, R_t. This may be accomplished by recursively subdividing the domain (cf. [26, 25]), as we illustrate for the region, R, given in Fig. 5a.

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

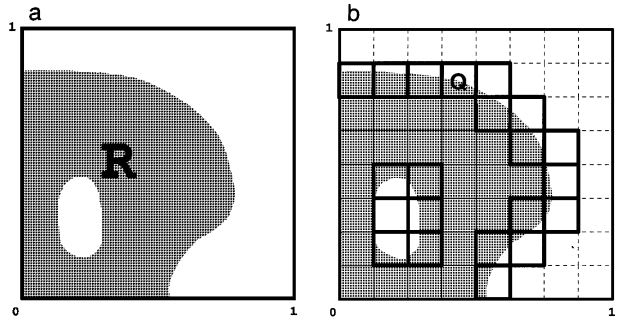

FIG. 5. Subdividing the domain into its coarsest subregions: (a) initial region, R; (b) coarsest subdivisions.

because the corresponding U-values can be evaluated in just $\mathcal{O}(n^2 \log(n))$ operations using a fast Fourier transform. 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 Fig. 5b 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;

$$\alpha_{ij} \iint_{\tilde{Q}} \cos(\pi i x) \cos(\pi j y)\, dA,$$

to each of the Fourier coefficients, c_{ij}, for $0 \le i, j \le n-1$. This case corresponds to the subregions of Fig. 5b 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 Fig. 5b.

Because Q is a mixed region, we divide it into quadrants, as shown in Fig. 6b. 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 further subdivision is required. See Fig. 6c.

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 $k = 2, 3, 4$

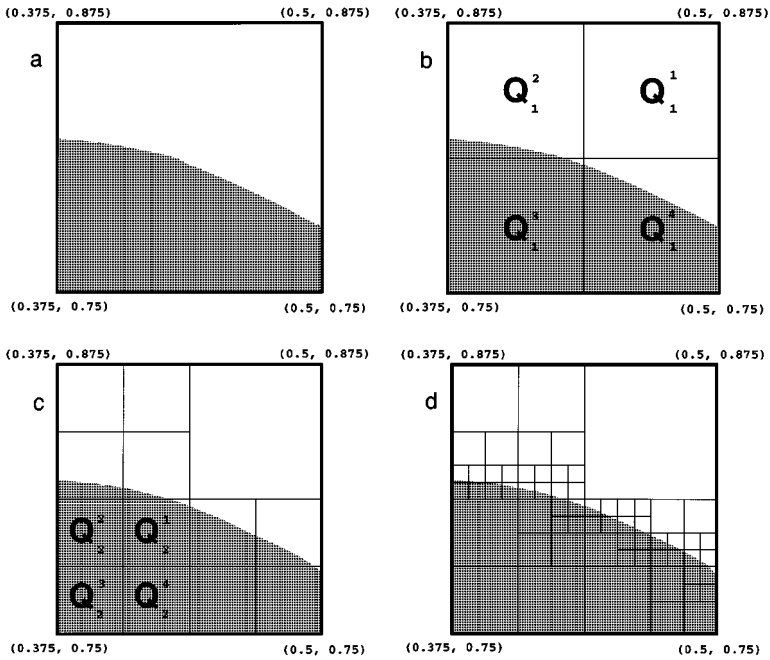

FIG. 6. Dividing a subregion: (a) initial subregion; (b) one subdivision; (c) two subdivisions; (d) four subdivisions.

and add contributions,

$$\alpha_{ij} \int\int_{Q_2^k} \cos(\pi i x) \cos(\pi j y)\, dA,$$

to each of the Fourier coefficients, c_{ij}, for $0 \le i,\ j \le n-1$. Recursive subdivisions of the domain continue (see, e.g., Fig. 6d) until regions containing multiple phases can be safely approximated by some simple numerical technique.

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 of width h. This section discusses how to approximate the contributions to the Fourier coefficients at the finest grid subdivisions.

3.3.1. *Piecewise Linear Approximation for Two-Phase Problems*

To produce an $\mathcal{O}(h^3/\Delta t + h^2)$ 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 (7) 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. To simplify the implementation of Step 2, we begin by breaking the square subdomain into two triangles and consider each separately.

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.

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.

Case 2. If two corners are in R, then we represent the shape as the difference of shapes which are treated using Cases 1 and 3.

Case 3. If three corners are in R, then we assume that the entire subdomain belongs to R, and we approximate the integrals (7) over the entire subdomain.

We seek an estimate of the error produced by this step for a smooth curve. One source of error occurs when smooth curves are approximated by line segments. By Fig. 7, 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 an interpolation. In this case we expect an $\mathcal{O}(h^3/\Delta t)$ error, based on the one-dimensional studies given in [22].

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

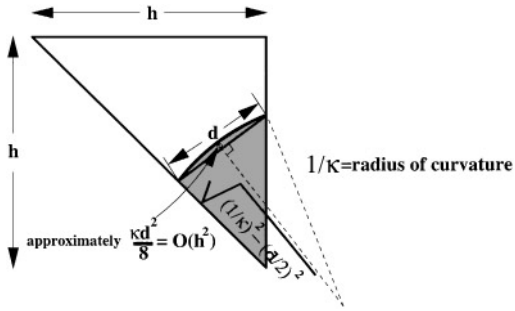

FIG. 7. Errors approximating curved segments.

Step 3. Integrate over each triangular subregion. We are now left with the task of adding a contribution

$$I_{ijk} = \alpha_{ij} \iint_{T_k} \cos(\pi i x) \cos(\pi j y)\, dA \tag{8}$$

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

Expanding the integrand about the centroid, $(\tilde{x}, \tilde{y})$, of T_k yields

$$I_{ijk} = \alpha_{ij}\, \text{Area}(T_k) \cos(\pi i \tilde{x}) \cos(\pi j \tilde{y}) + \mathcal{O}([i^2 + j^2]h^4) \tag{9}$$

where $\text{Area}(T_k)$ is the area of triangle T_k. This approximation is preferred over the direct evaluation of the integrals (8) 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.

Three-dimensional problems. The decomposition described above naturally extends to three dimensions [22]. First, each cube is divided into six tetrahedrons. The desired phase is then approximated with a number of small tetrahedrons. Finally, for each small tetrahedron, T_ℓ, a contribution

$$2^p\, \text{Volume}(T_\ell) \cos(\pi i \tilde{x}) \cos(\pi j \tilde{y}) \cos(\pi k \tilde{z})$$

is added to each Fourier coefficient, c_{ijk}, where $(\tilde{x}, \tilde{y}, \tilde{z})$ is the centroid of T_ℓ and p is the number of nonzero elements of $\{i, j, k\}$.

3.3.2. *Approximation of Junctions*

A number of methods for accommodating junctions are available [22]. A particularly simple and accurate approach is to recursively subdivide any region containing more than two phases. After only a few iterations, the smallest subregions that arise can be trivially treated by assigning an equal contribution to each set of Fourier coefficients.

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 $H = 1/n$.

3.4.1. *The Original Refinement Algorithm*

The original refinement algorithm of Section 3.2 is effective for a variety of problems. For certain smooth regions, however, small slivers of a region can be missed. Consider, for example, the shape found in Fig. 8. Applying the subdivision algorithm gives the mesh displayed in Fig. 9a. A close examination of the leftmost part of the shape indicates that a small, thin region is missed by the algorithm.

The original refinement algorithm also produces errors when applied to nonsmooth shapes. Consider, for example, the region displayed in Fig. 10. Such a shape may arise when a topological breaking occurs. Applying the original subdivision algorithm to the shape gives the mesh displayed in Fig. 11a. Clearly, an $\mathcal{O}(H^2)$ error in the phase area is produced at the cell containing the sharp corners. This corresponds to an $\mathcal{O}(\Delta t)$ error when $H = 1/n$ and n is chosen according to (5).

Although the errors produced by these flaws in the refinement technique often are less than those arising from the MBO-method, we prefer a more accurate refinement to achieve a greater confidence in our results. Furthermore, a more accurate refinement is required whenever higher order, extrapolated methods are used (see [21, 22]).

3.4.2. *A Method for a Gradual Refinement*

We now 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

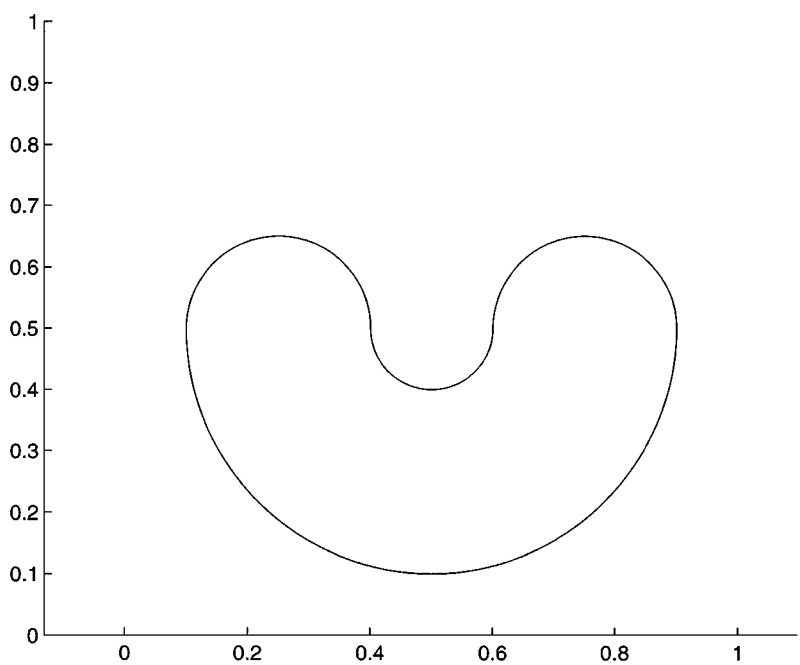

FIG. 8. A smooth region.

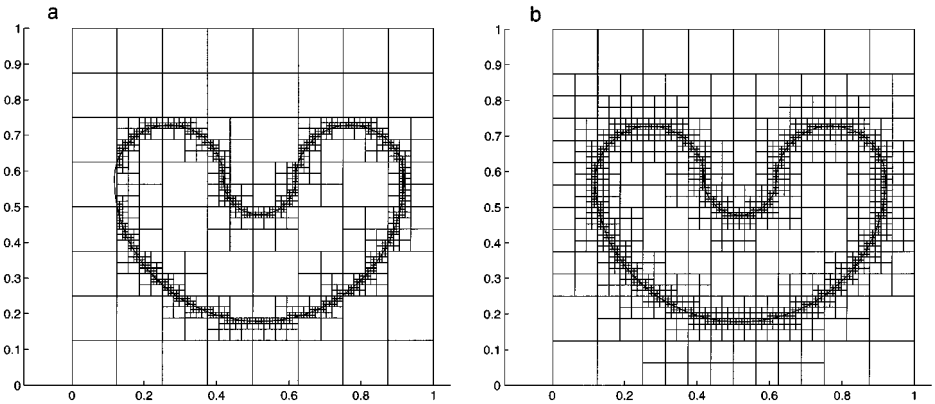

FIG. 9. Refinement methods for smooth regions: (a) original refinement; (b) gradual refinement.

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 Figs. 9b and 11b 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

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

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

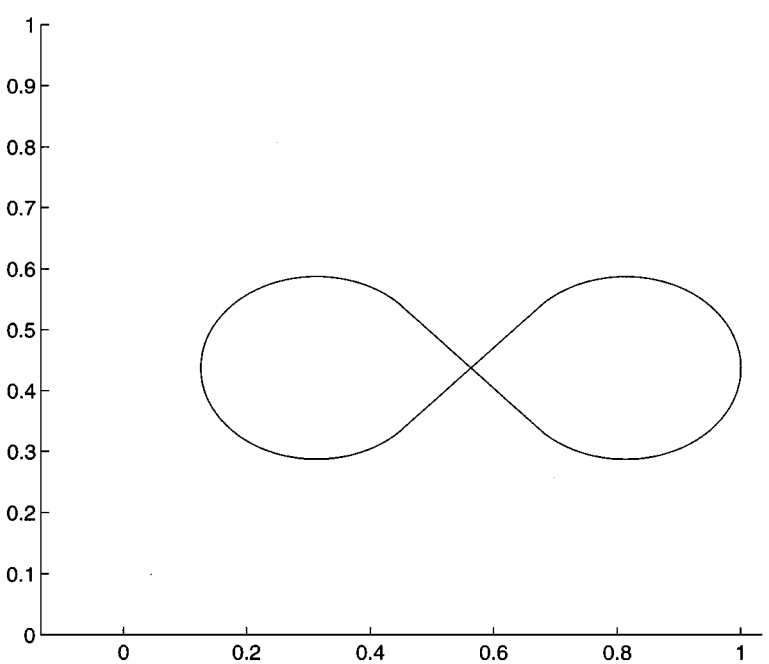

FIG. 10. A problem with sharp corners.

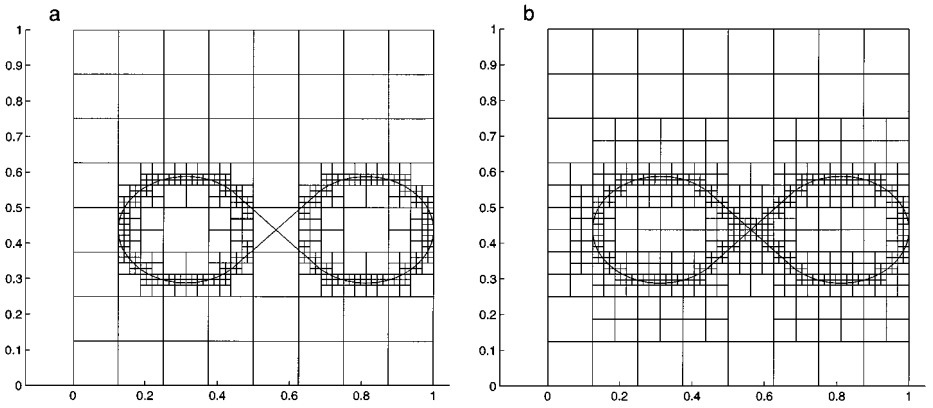

FIG. 11. Refinement methods for corners: (a) original refinement; (b) gradual refinement.

shorter than the perimeter of the domain). Letting $n_{\tilde{h}}$ be the number of cells of width $\tilde{h}$, we observe that

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

Thus, $\mathcal{O}(1/h + n^2)$ 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 [26, 25]. Our implementations define the grid as a list of vertices (cf. [8]) and access the cells and their neighbors indirectly by traversing their vertices [22].

3.5. Fast, Transform-Based Algorithms

The refinements of the previous sections lead to a large number of function evaluations,

$$U(x, y) = \sum_{i,j=0}^{n-1} c_{ij} \exp(-\pi^2[i^2 + j^2]\Delta t) \cos(\pi i x) \cos(\pi j y). \tag{10}$$

These evaluations occur at each vertex when subdividing the domain (see Section 3.2). Because these evaluations occur on an unequally spaced grid, a fast Fourier transform cannot be used. Letting the number of function evaluations be N_p, we see that direct evaluation of Eq. (10) is often prohibitively expensive because $\mathcal{O}(n^2 N_p)$ operations are required for each step of the algorithm. Similarly, evaluation of the Fourier coefficients using Eqs. (7) and (9) leads to a Fourier sum of the form

$$c_{ij} = \sum_{\ell=0}^{N_q-1} d_\ell \cos(\pi i x_\ell) \cos(\pi j y_\ell), \tag{11}$$

where $0 \le i,\ j \le n-1$, and (x_ℓ, y_ℓ) are unequally spaced. Here, N_q equals the number of vertices *inside* the region plus the number of triangles used to represent the region. Once again, a fast Fourier transform cannot be used and direct evaluation leads to $\mathcal{O}(n^2 N_q)$ operations. (Note that N_p and N_q are typically of the same magnitude.)

Several methods for the fast evaluation of Eqs. (10) and (11) have been developed [3, 6, 28]. In [3], for example, an efficient and practical method based on multiresolution analysis was developed that evaluates Eq. (10) at N_p points in

$$\mathcal{O}\left(N_p \log^2\left(\frac{1}{\epsilon}\right) + n^2 \log(n)\right) \tag{12}$$

operations and that evaluates all n^2 Fourier coefficients of (11) in

$$\mathcal{O}\left(N_q \log^2\left(\frac{1}{\epsilon}\right) + n^2 \log(n)\right) \tag{13}$$

operations, where ϵ is the precision of the computation.

Using the fact that $\mathcal{O}(1/h)$ refined cells arise in two dimensions (see Section 3.4.2), it is clear that $N_p = \mathcal{O}(1/h)$ and $N_q = \mathcal{O}(1/h)$. 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 $h \ll 1/n$, we see that a total of

$$\mathcal{O}((1/h)\log^2(h) + n^2 \log(n))$$

operations arise at each iteration of the spectral discretization of the MBO-method. As was shown in [22], the MBO-method produces an $\mathcal{O}(\Delta t^2)$ error in the position of a smooth curve at each step of the method when two phase regions occur. To avoid degrading this accuracy (see Section 3.3.1), we select $h = \mathcal{O}(\Delta t)$ to arrive at

$$\mathcal{O}\left(\frac{1}{\Delta t}\log^2(\Delta t)\right) \tag{14}$$

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

$$\mathcal{O}\left(\frac{1}{\Delta t}\log(\Delta t)\right) \tag{15}$$

operations are required per step to avoid degrading the overall accuracy of the method [22].

4. COMPARISON TO THE USUAL FINITE DIFFERENCE DISCRETIZATION OF THE MBO-METHOD

There are several reasons why the spectral method described in this article is preferred over the usual finite difference discretization of the MBO-method. These reasons are outlined below.

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

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

3. Local refinement is much simpler to implement for our 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 Δt practical (see [22, 21]), which in turn allows for larger time steps. When larger time steps are taken, even fewer basis functions are required to solve the heat equation to a given accuracy. Note that if extrapolation is used to eliminate the leading order error term, the operation counts (14) and (15) will no longer apply. To avoid degrading the accuracy of these (potentially) higher order methods requires $\mathcal{O}((\Delta t)^{-3/2}\log^2(\Delta t))$ operations in the two-phase case and $\mathcal{O}((\Delta t)^{-1}\log^2(\Delta t))$ operations in the multiphase case.

5. The original finite difference algorithm must satisfy (2) globally, or part of the front may erroneously remain stationary. By recursively refining near the interface and interpolating at the finest cell level, our approach eliminates this restriction.

6. The proposed spectral method also gives an $\mathcal{O}(h^3/\Delta t + h^2)$ 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

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

operations are needed per step for the basic method. This compares very favorably to the idealized finite difference result for smooth curves, $\mathcal{O}(1/\Delta t^3)$, which was derived in Section 2.4.

These are indeed formidable advantages for our proposed method over the usual finite difference discretization of the MBO-method. To illustrate the performance improvement, consider the motion by curvature of the kidney-shaped region displayed in Fig. 12. Using the new, spectral method and an optimized finite difference approach,[2] we compare the area lost over a time $t = 0.0125$ with the exact answer, $0.0125 \times 2\pi$ (see [17]). From Table I, we see that our proposed method can easily find solutions to within a 1% error. The finite difference approach, however, becomes impractical when accurate solutions are sought (see Table II).

Numerical tests for the problems described in the next section also found that our proposed discretization often requires less than 0.1% of the computational time of the usual finite difference discretization of the MBO-method. For this reason, the numerical studies in the following section are carried out using the new spectral method.

5. NUMERICAL EXPERIMENTS

In this section we report on various experiments using our algorithm. For further quantitative studies for both mean curvature flow and other, more general motions, see [22–24].

[2] The difference algorithm uses an adaptive time-stepping method on a uniform mesh. A multigrid technique was used to solve the implicit equations which arose from a backward Euler time-stepping scheme; see [22].

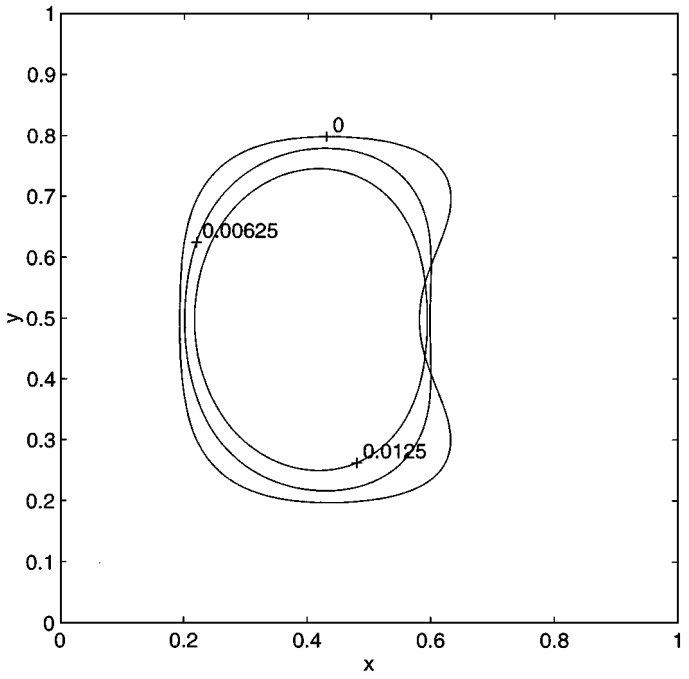

FIG. 12. A smooth interface at time, t.

5.1. A Smooth Three-Phase Problem

To begin, consider the motion by curvature of the three-phase problem given in Fig. 13. Using our new spectral method, the change in the area of the central region was compared to the exact result, $\frac{1}{2} \times (\pi/3)(2-6) \times 0.04$, which was obtained using the Von Neumann–Mullins parabolic law [17]. Because an $\mathcal{O}(\sqrt{\Delta t})$ error seems plausible from the asymptotic results given in [22, 23], an extrapolation in the area, $(1/(\sqrt{2}-1))(\sqrt{2}A^{\Delta t} - A^{2\Delta t})$, was also computed to eliminate the conjectured leading order error term. The results for a number of experiments are given in Table III.

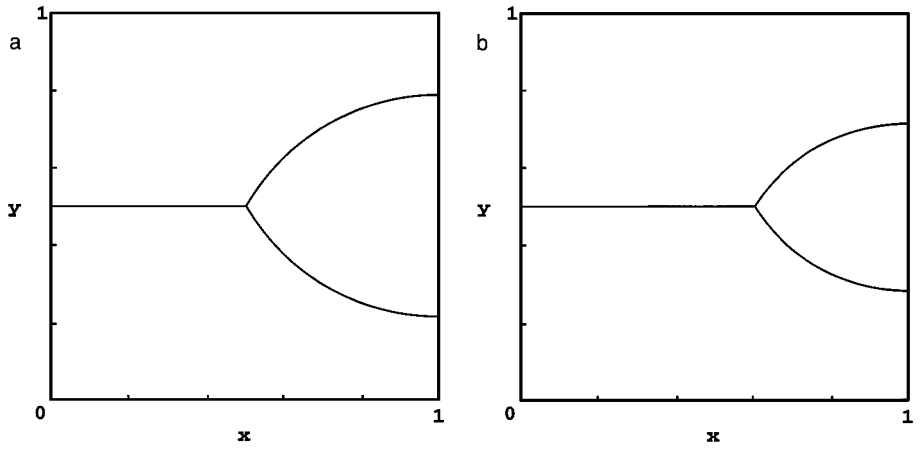

FIG. 13. A smooth three-phase problem: (a) $t = 0.0$; (b) $t = 0.04$.

TABLE I
New Spectral Method

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

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

TABLE II
Finite Difference Discretization

Δx	Error	Time
$\frac{1}{128}$	4%	85 s
$\frac{1}{512}$	3%	10341 s

TABLE III
Convergence Study for a Smooth Three-Phase Problem

Δt	h	$\overline{N_p}$	$\overline{N_q}$	*Error in* $A^{\Delta t}$	*Conv. Rate*	*Error in Extrapolation*	*Conv. Rate*
0.0025	2^{-11}	8.0e+03	1.3e+04	2.60e−03	0.67	−1.08e−03	1.33
0.00125	2^{-12}	1.6e+04	2.6e+04	1.70e−03	0.61	−4.63e−04	1.22
0.000625	2^{-12}	1.6e+04	2.6e+04	1.14e−03	0.57	−2.12e−04	1.13
0.0003125	2^{-13}	3.2e+04	5.3e+04	7.79e−04	0.55	−1.00e−04	1.08

Note. Here $\overline{N_p}$ and $\overline{N_q}$ represent the average values of N_p and N_q (see Section 3.5) that arise using a finest cell width of h (see Section 3.3). The value of n is 64 (see Section 3.1). All reported errors are independent of further cell refinement or larger values of n.

TABLE IV
Convergence Study for the Disappearance Time of a Phase Region

Δt	h	$\overline{N_p}$	$\overline{N_q}$	*Error in* $T^{\Delta t}$	*Conv. Rate*
0.0025	2^{-9}	1.7e+03	1.9e+03	4.61e−02	0.46
0.00125	2^{-9}	1.7e+03	1.9e+03	3.34e−02	0.46
0.000625	2^{-10}	3.1e+03	4.0e+03	2.42e−02	0.46
0.0003125	2^{-10}	3.le+03	3.8e+03	1.75e−02	0.46

Note. Here, $n = 64$. All reported errors are independent of further cell refinement or larger values of n.

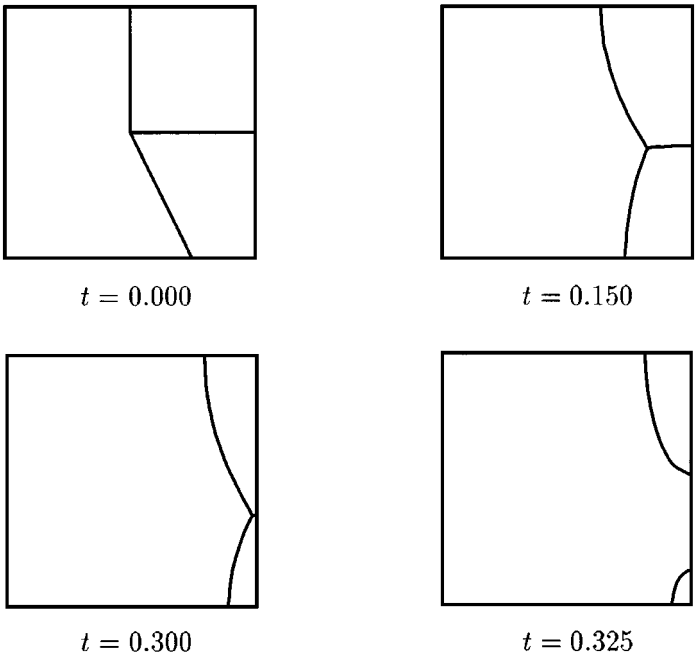

FIG. 14. The evolution of a junction through a singularity: (a) $t = 0.000$; (b) $t = 0.150$; (c) $t = 0.300$; (d) $t = 0.325$.

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

Note. The error arising from our spectral discretization represents less than 1% of the total error in the extrapolated results. This choice of error tolerance was made to study the convergence rate of the MBO-method and its extrapolation. Often, however, it is satisfactory to generate spatial errors which are comparable in size to the time-stepping errors generated by the MBO-method. For this choice of error tolerance, we found that fewer refinements were needed and the values of $\overline{N_p}$ and $\overline{N_q}$ were about one-tenth of those reported in Table III.

5.2. The Evolution of a Junction through a Singularity

The evolution of more complicated problems may also be simulated using our new, spectral discretization. Consider, for example, the motion by curvature of the three-phase problem given in Fig. 14. Using our spectral discretization of the MBO-method, estimates of the disappearance time, $T^{\Delta t}$, of the smallest phase were compared to the exact answer[3] for several Δt. The results for a number of experiments are reported in Table IV.

These results are suggestive of an approximately $\mathcal{O}(\sqrt{\Delta t})$ error for the basic method.

5.3. A Three-Dimensional Example

Interesting examples in three dimensions are also naturally handled by the method. For example, Fig. 15 displays the motion of a thin-stemmed barbell using a constant step size, $\Delta t = 0.0004$. From these images, it is clear that the center handle pinches off to form

[3] This result, $T = 0.33051$, was obtained using Brian Wetton's front tracking code; see [5].

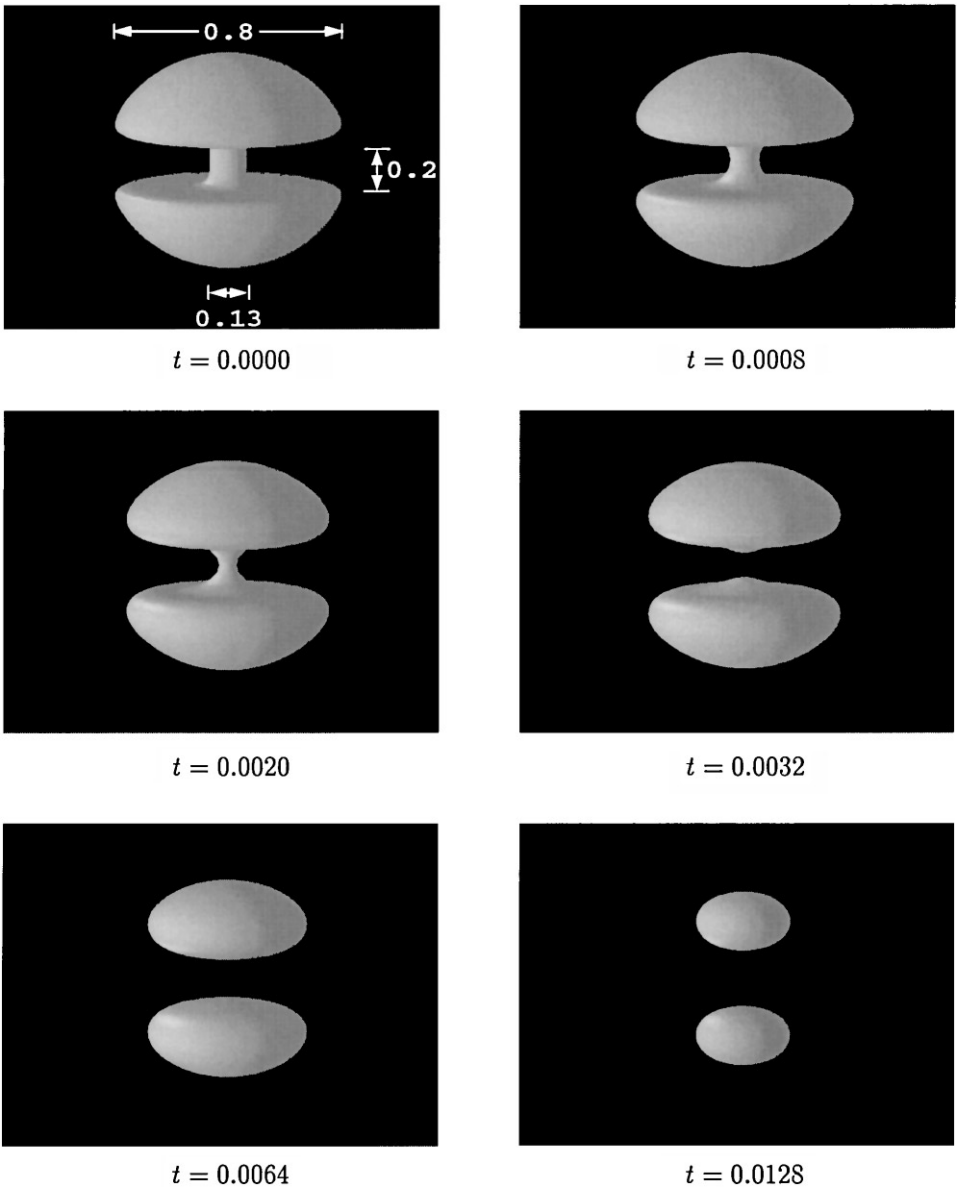

FIG. 15. A thin-stemmed barbell moving by mean curvature motion.

two pieces. As expected from [12], these convex shapes become nearly spherical as they disappear. This simulation used a constant step size, $\Delta t = 0.0004$ and required about 20 min of CPU time.

Note that a wider stem can produce a qualitatively different motion. For example, [22] gives the motion of a thick-stemmed barbell which exhibits no topological shape changes and eventually becomes ellipsoidal and more spherical as it disappears.

5.4. A Multiple Phase Example in Three Dimensions

The evolution of multiple phase junctions may also be studied using our new spectral discretization of the MBO-method. For example, Fig. 16 displays the motion by mean

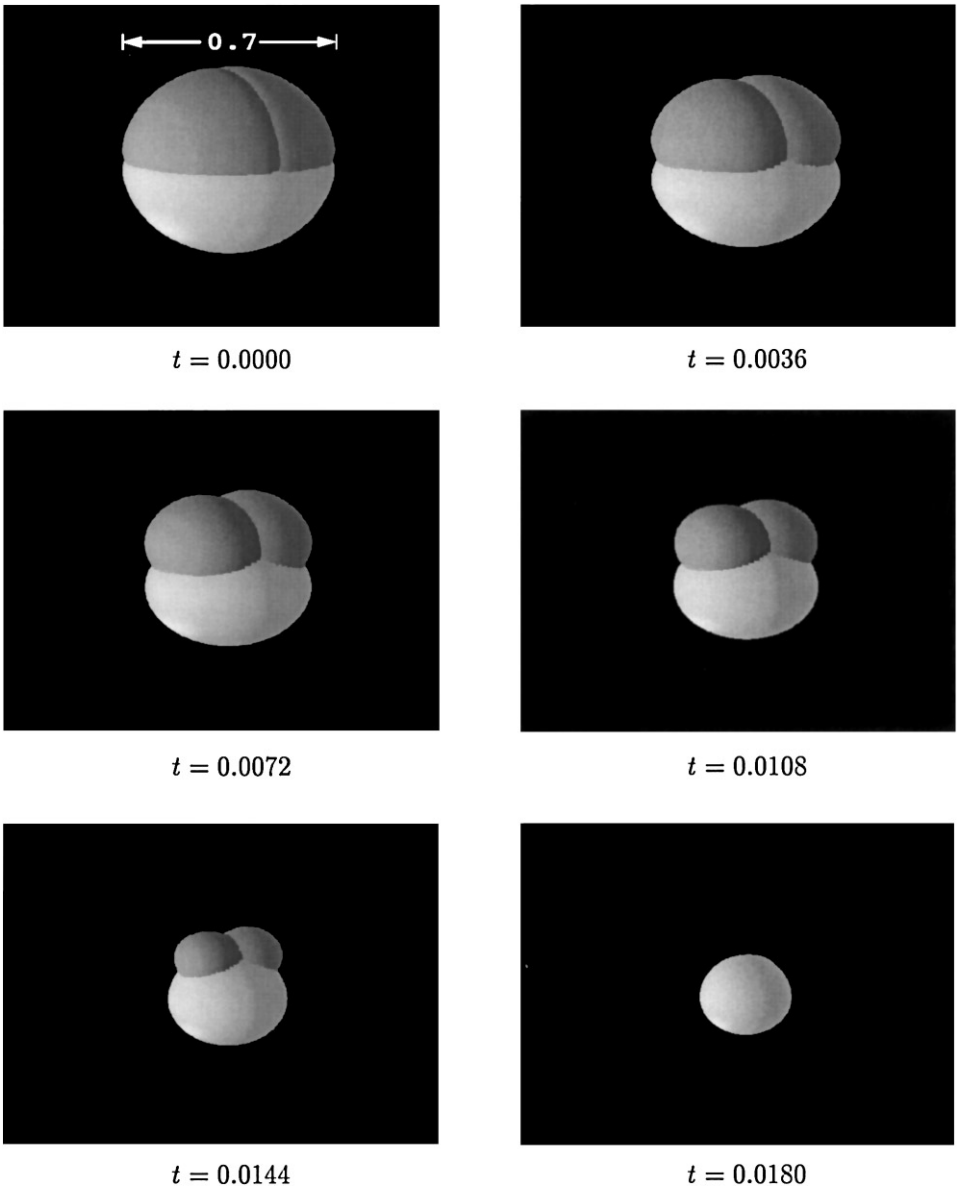

FIG. 16. A four-phase example moving by mean curvature motion.

curvature of a spherical four-phase shape. 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 [11]. This simulation used a constant step size, $\Delta t = 0.0004$ and required about 45 min of CPU time.

ACKNOWLEDGMENTS

Many thanks to Brian Wetton and Uri Ascher for helpful discussions. I also thank Leslie Greengard for suggesting the use of the unequally spaced fast Fourier transform.

REFERENCES

1. S. Angenent and M. E. Gurtin, Multiphase thermomechanics with interfacial structure 2. Evolution of an isothermal interface, *Arch. Rat. Mech. Anal.* **108**, 323 (1989).
2. G. Barles and C. Georgelin, A simple proof of convergence for an approximation scheme for computing motions by mean curvature, *SIAM J. Numer. Anal.* **32**(2), 484 (1995).
3. G. Beylkin, On the fast Fourier transform of functions with singularities, *Appl. Comput. Harmonic Anal.* **2**, 363 (1995).
4. 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 (1993).
5. L. Bronsard and B. T. R. Wetton, A numerical method for tracking curve networks moving with curvature motion, *J. Comput. Phys.* **120**(1), 66 (1995).
6. A. Dutt and V. Rokhlin, Fast Fourier transforms for nonequispaced data, *SIAM J. Sci. Statist. Comput.* **14**(6), 1368 (1993).
7. L. C. Evans, Convergence of an algorithm for mean curvature motion, *Indiana Univ. Math. J.* **42**, 553 (1993).
8. 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, Lisbon, Portugal, 1984*, p. 299.
9. J. Gravner and D. Griffeath, Threshold growth dynamics, *Trans. Am. Math. Soc.* **340**(2), 837 (1993).
10. J. W. Grove, Applications of front tracking to the simulation of shock rarefactions and unstable mixing, *Appl. Numer. Math.* **14**, 213 (1994).
11. D. Harker and E. R. Parker, Grain shape and grain growth, in *Trans. of the A.S.M., Cleveland, 1945*, p. 156.
12. G. Huisken, Flow by mean curvature of convex surfaces into spheres, *J. Differential Geom.* **20**, 237 (1984).
13. H. Ishii, A generalization of the Bence, Merriman and Osher algorithm for motion by mean curvature, in *Curvature Flows and Related Topics*, edited by A. Damlamian, J. Spruck, and A. Visintin (Gakkôtosho, Tokyo, 1995), p. 111.
14. H. Ishii, G. E. Pires, and P. E. Souganidis, Threshold dynamics type schemes for propagating fronts, *TMU Mathematics Preprint Series* 4, 1996.
15. B. Merriman, J. Bence, and S. Osher, Diffusion generated motion by mean curvature, in J. E. Taylor, editor, *Computational Crystal Growers Workshop*, edited J. E. Taylor (Am. Math. Soc., Providence, RI, 1992), p. 73.
16. B. Merriman, J. Bence, and S. Osher, Motion of multiple junctions: A level set approach, *J. Comput. Phys.* **112**(2), 334 (1994).
17. W. W. Mullins, Two-dimensional motion of idealized grain boundaries, *J. Appl. Phys.* **27**(8), 900 (1956).
18. S. Osher and J. A. Sethian, Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton–Jacobi formulations, *J. Comput. Phys.* **79**, 12 (1988).
19. S. Osher and C.-W. Shu, High-order essentially nonoscillatory schemes for Hamilton–Jacobi equations, *SIAM J. Numer. Anal.* **28**(4), 907 (1991).
20. 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.
21. 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*, p. 82.
22. S. J. Ruuth, *Efficient Algorithms for Diffusion-Generated Motion by Mean Curvature*, Ph.D. thesis, University of British Columbia, Vancouver, Canada, 1996.
23. S. J. Ruuth, A diffusion-generated approach to multiphase motion, CAM Report 97-28, University of California, Los Angeles, 1997.
24. S. J. Ruuth and B. Merriman, Convolution generated motion and generalized Huygens' principles for interface motion, CAM Report 98-4, University of California, Los Angeles, 1998.
25. H. Samet, *Applications of Spatial Data Structures: Computer Graphics, Image Processing, and GIS* (Addison–Wesley, Reading, MA, 1990).
26. H. Samet, *The Design and Analysis of Spatial Data Structures* (Addison–Wesley, Reading, MA, 1990).

27. C. S. Smith, Grain shapes and other metallurgical applications of topology, in *Metal Interfaces* (Am. Soc. Metals, Cleveland, 1952), p. 65.
28. E. Sorets, Fast Fourier transforms of piecewise constant functions, *J. Comput. Phys.* **116**, 369 (1995).
29. J. E. Taylor, J. W. Cahn, and C. A. Handwerker, I-Geometric models of crystal growth, *Acta Metall. Meter.* **40**(7), 1443 (1992).
30. H. Zhao, T. Chan, B. Merriman, and S. Osher, A variational level set approach to multiphase motion, *J. Comput. Phys.* **127**, 179 (1996).