JOURNAL OF COMPUTATIONAL PHYSICS **135,** 8–29 (1997)
ARTICLE NO. CP975721

A Simple Level Set Method for Solving Stefan Problems*

S. Chen,† B. Merriman,‡ S. Osher,‡ and P. Smereka§

†*Department of Mathematics, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213;* ‡*Department of Mathematics, University of California, Los Angeles, California 90095;* §*Department of Mathematics, University of Michigan, Ann Arbor, Michigan 48109*

Received July 24, 1996; revised March 18, 1997

A simple level set method for solving Stefan problems is presented. This method can be applied to problems involving dendritic solidification. Our method consists of an implicit finite difference scheme for solving the heat equation and a level set approach for capturing the front between solid and liquid phases of a pure substance. Our method is accurate with respect to some exact solutions of the Stefan problem. Results indicate that this method can handle topology changes and complicated interfacial shapes and that it can numerically simulate many of the physical features of dendritic solidification.

1. INTRODUCTION

In this paper, a numerical method is presented for solving Stefan problems and for simulating the behavior that arises from the unstable solidification of pure substances. This method accurately computes the boundary between the solid and liquid phases of a material as it undergoes the process of solidification, as well as the temperature of the material as it evolves over time.

Stefan problems typically involve the evolution of smooth boundaries or interfaces between different phases of a pure substance. For example, a Stefan model can be used to model the melting of ice in water. Problems such as these, as well as problems involving the stable solidification of a substance, are known as classical Stefan problems. In one dimension, Stefan problems have been studied in depth, and there are many excellent numerical algorithms for solving them. (See, for example, [2, 22].) However, the drawback for many of these methods is the difficulty one encounters when trying to extend these methods to higher dimensions.

Oftentimes, the goal of studying and developing algorithms for solving Stefan problems is to adapt and apply these methods to the problem of modeling unstable or dendritic solidification. The supercooled Stefan problem, coupled with an anisotropic curvature and velocity dependent boundary condition, is a model for unstable solidification. In particular, this model can be applied to the widely studied problem of dendritic crystal growth.

The process of crystal growth begins when one places a small seed of solid material into a surrounding bath of undercooled liquid. Heat is released at the solid/liquid interface into the liquid region. Undercooling drives the growth of the solid and triggers the instability mechanism, thereby causing the solid phase of the material to grow into the liquid phase in a fingerlike or dendritic fashion. The resulting interfacial shapes can be complex and thus, difficult to simulate numerically. However, accurate numerical algorithms are needed because they may be used to further our understanding of the role certain mechanisms play in governing crystal growth. Currently, the effect parameters such as anisotropy and surface tension have upon the shape of the crystal is of great interest to those involved in dendritic growth theory because these parameters are believed to determine the unique dendritic shape of the crystal. Hence these schemes could be of great value to scientists involved in such disciplines as chemistry, geology, physics and especially materials science, where controlling solidification is a fundamental goal (see [29]).

Before describing some of the numerical approaches for modeling dendritic solidification, we briefly review the physics of the problem. As described in [12], planar solidification fronts are morphologically unstable. This instability was first analyzed by Mullins and Sekerka [16]. In [6], Ivantsov found steady state solutions to the free boundary problem when the temperature at the interface is equal to a constant, i.e. when capillary and interfacial kinetic effects are disregarded. As outlined in [12], for any undercooling <1, there exists a whole family of solutions for a paraboloidal interface. From Ivantsov's solutions, we get a relationship between the given undercooling and the Peclet number, which is proportional to the product of the velocity, V, and the radius, R, of the dendritic tip.

Ivantsov's relation, $VR = \text{const}$, only provides a relation between V and R for a given undercooling. Consequently, much analytical work has been done in an attempt to understand how a unique dendritic operating state is selected for a fixed undercooling. Two major theories have

* Research supported by ARPA/ONR-N00014-92-J-1890, NSF DMS94-04942, and ARO DAAH04-95-1-0155.

0021-9991/97 $25.00

emerged: marginal stability theory and microscopic solvability theory. As described in [13], the marginal stability theory of Langer and Muller-Krumbhaar hypothesizes that the dendritic tip radius is the marginally stable wavelength from the Mullins–Sekerka analysis. From this, the theory predicts a unique operating state based upon a selection parameter.

More recently, microscopic solvability theory has been developed based upon solutions to the Nash–Glicksman integral equation. This theory states that there is no steady-state solution when the interfacial temperature is dependent upon isotropic surface tension. A stable stationary solution does exist, however, when surface tension is considered anisotropic. This theory leads to a solvability condition which in turn predicts a unique value for the tip radius. At present, it is unclear which theory is more accurate in predicting the unique dendritic operating state. We refer the interested reader to [12, 19] for a more thorough review of the underlying physics of modeling dendritic solidification.

What is clear from dendrite growth theory is that surface tension and anisotropy play an important role in determining the evolution of solidification fronts. Thus, any useful numerical method for modeling unstable solidification must be able to simulate anisotropic surface tension as well as other relevant physical parameters. Furthermore, it is advantageous for a numerical method to be able to simulate the intricate interfacial geometry that arises from crystal growth. Lastly, with the advent of supercomputers, there is considerable motivation to develop feasible numerical algorithms which translate easily to three dimensions.

Different numerical approaches for simulating crystal growth are often based on different formulations of the problem. For example, boundary integral methods are based upon numerically solving an integral equation on the moving boundary, i.e. the front. One drawback to boundary integral methods is that the necessary parametrization of the boundary makes it hard to extend such methods to higher dimensions. In one dimension, however, boundary integral methods work well and in [2] Brattkus and Meiron have obtained accurate results. Another approach has been to use finite element methods [17, 23]. Although these adaptive algorithms tend to be rather complicated and computationally expensive, the three-dimensional simulations in [23] are impressive and prove that finite element methods are competitive with other numerical approaches.

Employing front-tracking methods has always been a common way of solving moving boundary problems. In [7], Juric and Tryggvason presented a numerical method which incorporated ideas from the immersed boundary method for transferring information from the moving boundary to the fixed temperature grid. Their method was successful in modeling many of the physical features of dendritic solidification, such as discontinuous material properties. However, special care had to be taken (as is common for front-tracking methods) when topological changes such as merging occurred at the front. Another front-tracking method was introduced by Roosen and Taylor [21]. They assumed that the shape of the crystal was a polygon and thus were able to avoid directly computing curvature at the front. Unlike traditional front-tracking methods, their method was able to detect and deal with topological changes. However, their method did suffer from a grid induced anisotropic effect. In [1], Almgren also used a method that explicitly tracks the interface, along with a new variational algorithm for computing dendritic solidification. His formulation was based on relating the Gibbs–Thomson relation to a local equilibrium condition in which an energy functional, dependent upon bulk and surface energies, is minimized. With this method. Almgren performed many quantitative experiments comparing the numerical results with what is predicted from dendritic growth theory. When comparing the radii and velocities of dendritic tips, his results were close to the Ivantsov solutions.

Phase-field methods have become increasingly popular over the past few years. These methods are based on phase-field models, which differ from the classical model of a sharp interface. In a phase-field model, the boundary is ''spread out'' and a phase-field variable ϕ is introduced such that away from the boundary, $\phi = 0$ or 1, and at points on the interface, $\phi \in (0, 1)$. The equations of motion are recast in terms of ϕ and the location of the front is stored implicitly in ϕ. By using a phase-field approach, interfacial geometric quantities such as curvature and the outward normal vector do not have to be computed since they are already included in the model. Phase-field methods present an advantage over front-tracking methods because complex interfacial shapes pose no problem since the front is not being explicitly tracked.

There have been many papers published about phase-field models and related phase-field methods (see [3, 4, 8–10, 20, 28]). Recently, in [28], Wang and Sekerka used a thermodynamically consistent phase-field model to construct numerical algorithms, which they used to study the morphologies of dendritic tips. Similarly in [9], Karma and Rappel presented a phase-field method which yielded numerical results in close agreement with steady-state solutions. Their method is based on an analysis which expands the range of applicability of the phase-field method to include smaller kinetic effects and smaller ratios of capillary length to interface thickness. Based on this analysis, they have been able to apply a phase-field method to simulate three-dimensional dendritic growth ([8]).

There is an inherent disadvantage to using phase-field methods. Roughly speaking, the evolution equation for ϕ takes the form of a reaction–diffusion equation, i.e., $\phi_t =$

$\varepsilon\,\Delta\phi - (1/\varepsilon)F(\phi)$, where ε is a parameter that depends upon the interface thickness. In [15], Merriman, Bence, and Osher proved that to numerically resolve such an equation, the mesh size Δx is restricted by the relation $(\Delta x/\varepsilon) \lll 1$. So although in theory, solutions to phase-field model equations converge to the solutions of sharp interface model equations as $\varepsilon \to 0$, numerically speaking, phase-field methods are held back by their inability to resolve the interface properly. For example, many features of the three-dimensional dendritic simulations in [10] (which were based on a phase-field model) have since been shown to be mesh dependent.

Solidification problems are essentially problems involving moving boundaries and as such, they are well suited for numerical simulation by the level set method. Level set methods are ideal for moving boundary problems because instead of tracking the boundary or front using a Lagrangian approach; one can instead capture the front on a fixed grid (Eulerian approach). In [18], the level set method was first introduced. Since then, it has been applied to numerous problems (e.g., see [5, 15, 24, 27]). The main idea behind the level set method is that the front is always represented by the zero level set of a smooth, continuous function. Hence, the front can be graphed simply by plotting a specific contour level. One of the main advantages the level set method affords over front-tracking methods and boundary integral methods is that the front is never explicitly tracked and, hence, complicated interfacial shapes can be represented easily, including interfacial topology changes arising from the merging of two crystalline fronts. Also, the level set method can be easily extended to higher dimensions.

In [24], the authors first presented a somewhat complicated and computationally expensive level set method for solving problems involving crystal growth and dendritic solidification. Their method combined a level set approach with a boundary integral formulation of the problem. Our method differs from [24] in that we take a simpler approach. We avoid using a boundary integral method to compute the normal velocity at the interface. Also, following the results and work done in [27], we reinitialize the level set function to be signed distance function at every timestep. Our method improves upon the method in [24] because it retains all the advantages of using a level set approach without any of the complications and restrictions that arise from employing a boundary integral method. Furthermore, since our method is based upon a sharp interface model, it has an advantage over phase-field methods in the sense that the grid size is not constrained by an arbitrary parameter representing the thickness of the front.

In this paper, we present the details and results from this new numerical method. In Section 2, we outline the formulation of the Stefan problem and how we modify it to model unstable solidification. In Section 3, we present the algorithm and in Section 4, we discuss the details of its numerical implementation. Some of the results of using this method are shown in Section 5 and in Section 6 we draw conclusions.

2. EQUATIONS OF MOTION

As detailed in [14], the Stefan problem consists of finding the temperature and the boundary between different phases of a pure material. These two variables evolve by the diffusion of heat from external and internal heat sources. We are mainly concerned with the two-phase one-front Stefan problem for which there exist classical and generalized solutions.

As mentioned before, the modified Stefan problem with supercooled liquids, is an unstable problem that can be used to model the spontaneous pattern formation that arises in dendritic solidification. As described in [11], "this instability occurs because diffusion kinetics favors configurations in which the growing solid has as large a surface area as possible." Even with smooth initial interfacial shapes, the evolving front will cease to be smooth and can become quite complicated. Work done by Mullins and Sekerka on the stability analysis of this problem proved that a small perturbation to a flat interface will grow unstably in the case of supercooled temperatures.

We consider the two-phase Stefan problem. In the case of modeling dendritic solidification, we include effects of undercooling, crystalline anisotropy, surface tension, and molecular kinetics. We consider a square domain or box, D, of a pure material where at every timestep and at every gridpoint the material is either in liquid or solid phase. Let $T(\mathbf{x}, t)$ represent the temperature of the material. The region where the material is solid is denoted as Ω, and the region where the material is liquid as Ω^c. The interface between the solid and liquid phases, i.e., the boundary of Ω, will be denoted by Γ. Let V represent the normal velocity at the front Γ.

The governing equations for our formulation of the problem are

$$c_s \frac{\partial T}{\partial t} = \nabla \cdot (k_s \nabla T), \quad \mathbf{x} \in \Omega, \tag{1}$$

$$c_l \frac{\partial T}{\partial t} = \nabla \cdot (k_l \nabla T), \quad \mathbf{x} \in \Omega^c, \tag{2}$$

where c_s and c_l denote the volumetric heat capacities and k_s and k_l are the thermal diffusivities of the material in Ω and Ω^c, respectively. On Γ, the jump condition

$$LV = -\left[k_l \frac{\partial T_{\text{liq}}}{\partial \mathbf{n}} - k_s \frac{\partial T_{\text{sol}}}{\partial \mathbf{n}} \right] \tag{3}$$

holds, where L denotes the latent heat of solidification. The jump is taken from liquid to solid, and the vector $\mathbf{n}$ is the outward normal vector at the front. In the liquid region, $\partial T_{\text{liq}}/\partial \mathbf{n}$ denotes the normal derivative of T and $\partial T_{\text{sol}}/\partial \mathbf{n}$ the corresponding normal derivative of T in the solid region. Equation (3) is commonly referred to as the Stefan condition.

For a classical Stefan problem, one sets $T(\mathbf{x}, t) = T_m$ on Γ, where T_m is a constant equal to the melting temperature of the material. But for application to problems involving crystal growth and dendritic solidification, one would like to take into account the effects of surface tension, crystalline anisotropy, and molecular kinetics. Thus, the second boundary condition we consider is the classical Gibbs–Thomson relation; for $\mathbf{x} \in \Gamma$,

$$T(\mathbf{x}, t) = -\varepsilon_C \kappa - \varepsilon_V V, \tag{4}$$

where κ denotes the curvature at the front, ε_C is the surface tension coefficient, and ε_V is the molecular kinetic coefficient. In the isotropic case, both ε_C and ε_V are taken to be constant. For the anisotropic case, following the notation found in [24], one can take

$$\varepsilon_C(\mathbf{n}) = \overline{\varepsilon_C}(1 - A\cos(k_A\theta + \theta_o)) \tag{5}$$

$$\varepsilon_V(\mathbf{n}) = \overline{\varepsilon_V}(1 - A\cos(k_A\theta + \theta_o)), \tag{6}$$

where the constants A, k_A, θ_o, $\overline{\varepsilon_C}$, and $\overline{\varepsilon_V}$ depend upon the material. Here, θ is the angle between the x-axis and $\mathbf{n}$, while θ_o controls the angle of the symmetry axis upon which the crystal grows.

In all our experiments, we set the thermal diffusivities and heat capacities equal to one in both liquid and solid regions. Also, unless otherwise noted, we set $L = 1$. Thus, the somewhat simplified problem becomes that of finding $T(\mathbf{x}, t)$ and $\Gamma(t)$ such that equations

$$\frac{\partial T}{\partial t} = \Delta T, \qquad \mathbf{x} \in D\backslash\Gamma, \tag{7}$$

$$V = -\left[\frac{\partial T}{\partial \mathbf{n}}\right], \quad \mathbf{x} \in \Gamma(t), \tag{8}$$

are satisfied, along with Eqs. (4), (5), and (6).

3. DESCRIPTION OF ALGORITHM

Our method for solving the Stefan problem uses a level set approach to effectively capture the front at each new timestep, and an implicit finite difference scheme to solve the heat equation everywhere away from the front. This new method improves upon an earlier level set approach [24] because it does not keep track of the front's history of motion.

3.1. *Level Set Function and Related Equations*

We construct a level set function ϕ, such that at any time t, the front is equal to the zero level set of ϕ, i.e.,

$$\Gamma(t) = \{\mathbf{x} \in D : \phi(\mathbf{x}, t) = 0\}. \tag{9}$$

Initially, ϕ is set equal to the signed distance function from the front such that ϕ is positive in Ω^c (liquid phase) and negative in Ω (solid phase),

$$\phi(\mathbf{x}, 0) = \begin{cases} +d, & \mathbf{x} \in \Omega^c, \\ 0, & \mathbf{x} \in \Gamma, \\ -d, & \mathbf{x} \in \Omega, \end{cases}$$

where d is the distance from the front.

The idea behind the level set method is to move ϕ with the correct speed, V, at the front and then to update the temperature, $T(\mathbf{x}, t)$, with the new position of the front stored implicitly in ϕ. With this approach, we avoid any difficulties that come from explicitly tracking the front and we increase our ability to deal with complex interfacial shapes.

Given the normal speed, V, at which the front moves, we want to construct a speed function, F, which is a continuous extension of V off Γ onto all of D. The equation of motion governing ϕ is then given by

$$\phi_t + F|\nabla\phi| = 0. \tag{10}$$

This equation will move ϕ with the correct speed at the front so that Γ will always be equal to the zero level set of ϕ.

We also use ϕ to define the outward normal vector $\mathbf{n}$ by

$$\mathbf{n} = \nabla\phi/|\nabla\phi| \tag{11}$$

and the curvature term κ by

$$\kappa = \nabla \cdot \mathbf{n} = \nabla \cdot \left(\frac{\nabla\phi}{|\nabla\phi|}\right). \tag{12}$$

From Eqs. (8) and (11), we can rewrite the expression for V as

$$V = -[\nabla T] \cdot \mathbf{n} = -[\nabla T] \cdot \left(\frac{\nabla\phi}{|\nabla\phi|}\right), \tag{13}$$

where the jump in $[\nabla T]$ is taken from liquid to solid re-

gions. Since F is equal to V along the interface, we can combine Eqs. (10) and (13) to get the following equation, which of course is only valid on the zero level set of ϕ:

$$\phi_t = [\nabla T] \cdot \nabla \phi, \quad \mathbf{x} \in \Gamma. \tag{14}$$

Next, we need to extend the velocity function V in a reasonable way to a small two-dimensional region which includes Γ.

3.2. *Extension of Normal Velocity off the Interface*

In our algorithm, we compute approximations to $[\nabla T]$ at every gridpoint. The problem that arises in computing $[\nabla T]$ is that this quantity is $O(1)$ only at points close to or on the front. Let F be defined as an extension of V off of Γ; such an extension should not be discontinuous near the interface. By constructing F to be a continuous extension of V, we then avoid unnecessary numerical difficulties when we solve Eqs. (10) and (14).

The approximation to $[\nabla T]$ is based upon approximations to the derivatives of T in four coordinate directions (the standard x, y Cartesian coordinates and the 45°-rotated coordinates ζ and η). We use these four coordinate directions to cut down on grid orientation effects, as will be explained in more detail in Section 5.

Each approximation to the jump in a derivative of T can be continuously extended away from the front by the advection equations

$$u^1_t + S(\phi)\phi_x u^1_x = 0 \tag{15}$$

$$u^2_t + S(\phi)\phi_y u^2_y = 0 \tag{16}$$

$$u^3_t + S(\phi)\phi_\eta u^3_\eta = 0 \tag{17}$$

$$u^4_t + S(\phi)\phi_\zeta u^4_\zeta = 0, \tag{18}$$

where $u^1 = [\partial T/\partial x]$, $u^2 = [\partial T/\partial y]$, $u^3 = [\partial T/\partial \eta]$ and $u^4 = [\partial T/\partial \zeta]$ on Γ. S is equal to the sign function. Equations (15) through (18) have the effect of continuously extending u^1, u^2, u^3, u^4 away from the front by advecting these fields in the proper upwind direction. Note that these equations will not degrade the value of V on the front because ϕ is zero on Γ, hence, so are $S(\phi)\phi_x$, $S(\phi)\phi_y$, $S(\phi)\phi_\eta$, and $S(\phi)\phi_\zeta$.

3.3. *Reinitialization of ϕ*

From Eqs. (11), (12), and (13), we see that computation of the normal vector, normal velocity, and curvature at the front are all dependent upon the level set function ϕ. However, by Eq. (10), the level set function will cease to be an exact distance function even after one timestep. In order to keep the approximations to $\mathbf{n}$, V, and κ accurate, we want to avoid having steep or flat gradients develop in ϕ. One way to avoid these numerical difficulties is to reinitialize ϕ to be an exact distance function from the evolving front Γ at each timestep.

The process we use to reinitialize ϕ is due to work and results found in [27]. In that paper, an algorithm was presented for reinitializing the level set function ϕ to be an exact signed distance function from the front. The basic idea behind this method is that given a function ϕ_0 that is not a distance function, one can evolve it into a function ϕ that is an exact signed distance function from the zero level set of ϕ_0. This is accomplished by iterating the equation

$$\phi_t = S(\phi_0)(1 - |\nabla \phi|) \tag{19}$$

to steady state, where $\phi(\mathbf{x}, 0) = \phi_0(\mathbf{x})$ and S again denotes the sign function. As in [27], we smooth the sign function S by the equation

$$S_\varepsilon(\phi_0) = \frac{\phi_0}{\sqrt{\phi_0^2 + \varepsilon^2}} \tag{20}$$

to avoid any numerical difficulties.

By using this approach, we avoid having to explicitly find the contour $\phi_0 = 0$ and then resetting values of ϕ_0 at gridpoints neighboring the front. From Eq. (19), it is clear that the original position of the front will not change, but at points away from Γ, ϕ will be evolved into a distance function.

3.4. *Updating the Temperature*

After moving ϕ by the correct velocity at the front and then reinitializing ϕ to be an exact signed distance function from Γ, we update the temperature T of the material. Updating T essentially boils down to solving the heat equation over the whole domain D, with special care taken at points near the interface between the two phases.

At points away from the front, we solve the heat equation using a standard five-point stencil. When we are at points near the front, we use one-sided differencing and values of ϕ to incorporate the front's position into the stencil. We thereby effectively capture the front using the level set function ϕ.

For points near Γ, we employ an interpolation scheme to approximate the spatial double derivatives of T. Since after reinitialization, ϕ is an exact distance function, we can use ϕ to detect when we are close to or on the front Γ. Moreover, we can use ϕ to interpolate the distances between a point on the front, $\mathbf{x}_f \in \Gamma$, and gridpoints neighboring it in either the vertical or horizontal direction. (Note we are only considering those points on the front which intersect some gridline $x = p\ \Delta x$ and/or $y = q\ \Delta y$.) If a gridpoint intersects the front, we set the value of T at that

gridpoint equal to the value given by the Gibbs–Thomson relation, Eq. (4).

For example, suppose $\mathbf{x}_f$ intersects some horizontal gridline, $y = J\,\Delta y$, for J equal to an integer. Then, using divided differences tables, one can interpolate two polynomials, P^{L} and P^{R}, which are constructed from $\mathbf{x}_f$ and gridpoints to the left and right of the front, respectively. P^{L} and P^{R} can then be differentiated twice to obtain the coefficients used to approximate T_{xx} at the two gridpoints bordering $\mathbf{x}_f$ in the horizontal direction.

One advantage to this method is that higher order accuracy can be achieved simply by using a higher order interpolant to T for points near the front. Higher order interpolants are constructed by adding more gridpoints to the divided differences tables. In the special case where two fronts are merging and there are not enough gridpoints in between to achieve standard second-order accuracy, we use a first-order interpolating polynomial. In general, this interpolation scheme makes the extension of this method to higher dimensions straightforward and easy.

3.4.1. *Curvature.* In two dimensions, the Gibbs–Thomson relation (Eq. (4)) governs the value of T on the front. Hence, the curvature, κ, at the front needs to be computed. From Eq. (12), κ in nonconservative form can be rewritten as

$$\kappa = \frac{(\phi_y^2\phi_{xx} - 2\phi_x\phi_y\phi_{xy} + \phi_x^2\phi_{yy})}{(\phi_x^2 + \phi_y^2)^{3/2}}. \tag{21}$$

We compute the value of κ at gridpoints neighboring the front, then we interpolate its value on the front whenever it is needed. Equation (21) is numerically solved using centered finite difference approximations to the partial derivatives of ϕ.

3.5. *Outline of the Method*

The steps of our algorithm can now be outlined as follows:

Step 1. Initialize $T(\mathbf{x}, t)$ and $\phi(\mathbf{x}, t)$ so that ϕ is the signed normal distance from the interface between the two phases of the material.

Step 2. Compute the velocity field $F(\mathbf{x}, t)$, which is a continuous extension of the normal velocity V at the front onto the whole domain D. F is computed from approximations to $[\partial T/\partial \mathbf{n}]$.

Step 3. Update ϕ by the equation, $\phi_t + F|\nabla\phi| = 0$, for one timestep. The front's new position is now equal to the zero level set of ϕ. Denote this updated ϕ as ϕ_0. (Note that ϕ_0 is not a distance function.)

Step 4. Reinitialize ϕ to be an exact signed distance function by solving the equation, $\phi_t = S(\phi_0)(1 - |\nabla\phi|)$ to steady state. Here $\phi(\mathbf{x}, 0) = \phi_0(\mathbf{x})$.

Step 5. Away from Γ, solve for T by discretizing the heat equation using an implicit centered finite difference scheme. For gridpoints less than or equal to a stepsize away from the front, use ϕ to interpolate polynomials approximating T. These polynomials are constructed so that their values on the front satisfy the Gibbs–Thomson relation, Eq. (4). Differentiate these polynomials to obtain values of ΔT which can then be used to update T at those gridpoints neighboring Γ.

Step 6. Repeat Steps 2 through 5 to get the next updated values of ϕ and T.

4. DISCRETIZATION

In all of our computations, we take the domain D to be a square box. Both Δx and Δy are equal to a uniform mesh size h. For a given square side of length SQL, we set $h = SQL/M$, where $(M + 1)^2$ is the total number of gridpoints on the grid. The timestep taken in the main loop of the algorithm is Δt, i.e., the timestep taken when we discretize Eqs. (7) and (10). We take the following definitions throughout the rest of the section:

$$\mathbf{x}_{i,j} = ((i-1)h, (j-1)h)$$
$$\phi_{i,j} = \phi(\mathbf{x}_{i,j})$$
$$T_{i,j} = T(\mathbf{x}_{i,j})$$
$$i, j = 1, ..., M+1.$$

In this section, for convenience we assume that the square domain D is of the form $[0, SQL] \times [0, SQL]$, but in our experimental results (presented in Section 5), we often take D to be of the form $[-SQL/2, SQL/2] \times [-SQL/2, SQL/2]$.

4.1. *Computation of Normal Velocity Components*

We first compute approximations to the jump in ∇T across the interface. Breaking this down even further, we compute jumps in the derivatives of T in the aforementioned four coordinate directions. (See Fig. 1.) Thus, we compute approximations to $[\partial T/\partial x]$, $[\partial T/\partial y]$, $[\partial T/\partial \eta]$ and $[\partial T/\partial \zeta]$.

Using the same notation as in Section 3.2, we compute four fields: $u^1_{i,j}, u^2_{i,j}, u^3_{i,j}$, and $u^4_{i,j}$, which are defined on all of D. At gridpoints on or near Γ, $u^1_{i,j}, u^2_{i,j}, u^3_{i,j}$, and $u^4_{i,j}$ approximate the jumps in $\partial T/\partial x$, $\partial T/\partial y$, $\partial T/\partial \eta$, and $\partial T/\partial \zeta$ (respectively) across the interface. Away from Γ, u^1–u^4 are generally close to zero.

The discretizations we use to compute $u^1_{i,j}$, $u^2_{i,j}$, $u^3_{i,j}$, and $u^4_{i,j}$ are

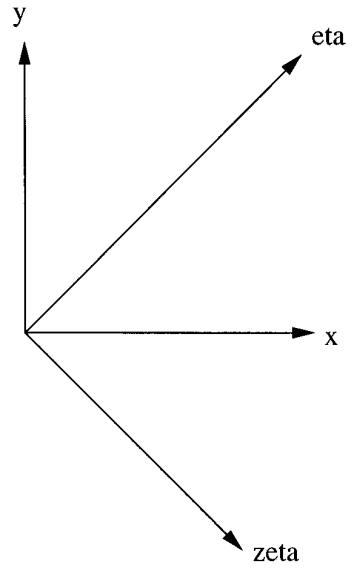

FIG. 1. Four coordinate directions used to compute the normal velocity.

$$u^1_{i,j} = -S_{i,j}(\phi_x)((T_{i+2,j} - T_{i+1,j}) - (T_{i-1,j} - T_{i-2,j}))/h$$
$$u^2_{i,j} = -S_{i,j}(\phi_y)((T_{i,j+2} - T_{i,j+1}) - (T_{i,j-1} - T_{i,j-2}))/h$$
$$u^3_{i,j} = -S_{i,j}(\phi_\eta)((T_{i+2,j+2} - T_{i+1,j+1}) - (T_{i-1,j-1} - T_{i-2,j-2}))/(\sqrt{2}h)$$
$$u^4_{i,j} = -S_{i,j}(\phi_\zeta)((T_{i+2,j-2} - T_{i+1,j-1}) - (T_{i-1,j+1} - T_{i-2,j+2}))/(\sqrt{2}h).$$

The sign functions of the different derivatives of ϕ in the above discrete equations are necessary in order to ensure that the jumps are consistently computed from solid to liquid phases.

4.2. *Discretization of the Velocity Extension*

As mentioned in Section 3.2, sharp jumps may develop in the computation of $u^1_{i,j}, u^2_{i,j}, u^3_{i,j}$, and $u^4_{i,j}$. These discontinuities may affect the accuracy of the calculation of the velocity field F and the updating of ϕ from Eq. (10). Accordingly, we would like to extend $u^1_{i,j}, u^2_{i,j}, u^3_{i,j}$, and $u^4_{i,j}$ from Γ to all of D.

We continuously extend V off the front by solving an appropriate advection equation for each component. For gridpoints on opposite sides of Γ, we want the characteristics for the advection equation to point in opposite directions, as shown in Fig. 2. Equations (15)–(18) were derived based on the fact that $S(\phi)\phi_x$, $S(\phi)\phi_y$, $S(\phi)\phi_\eta$, and $S(\phi)\phi_\zeta$ will properly control the direction in which we want the characteristics for each advection equation to point.

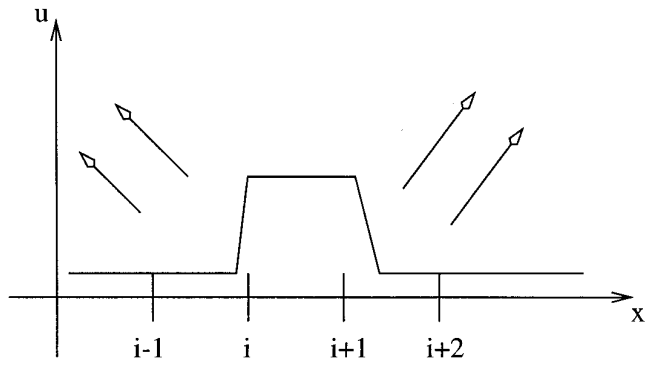

FIG. 2. Profile of u^1: The front is between x_i and x_{i+1}. When extending u^1, characteristics for u^1 should point in opposite directions.

We discretize Eqs. (15)–(18) with a first-order upwind scheme. The choice of the timestep Δt_{extend} is completely arbitrary and not necessarily related to the main timestep Δt. The only constraint we need to impose on the timestep Δt_{extend} is that it satisfy the CFL condition: $\Delta t_{\text{extend}}/h \le 1$. Thus we discretize Eq. (15) by the scheme

$$\text{if } S_{i,j}(\phi\phi_x) > 0, \text{ then } u^{1(\text{new})}_{i,j} = u^{1(\text{old})}_{i,j} - \text{cfl}^*(u^{1(\text{old})}_{i,j} - u^{1(\text{old})}_{i-1,j})$$
$$\text{if } S_{i,j}(\phi\phi_x) < 0, \text{ then } u^{1(\text{new})}_{i,j} = u^{1(\text{old})}_{i,j} + \text{cfl}^*(u^{1(\text{old})}_{i+1,j} - u^{1(\text{old})}_{i,j})$$

with cfl set to 0.5. Equations (16)–(18) are discretized similarly.

4.3. *Discretization in Time*

When we solve for Eqs. (10) and (19), we need to compute approximations to the spatial derivatives of ϕ. For increased accuracy we use second-order ENO approximations. To avoid any instabilities arising from the temporal

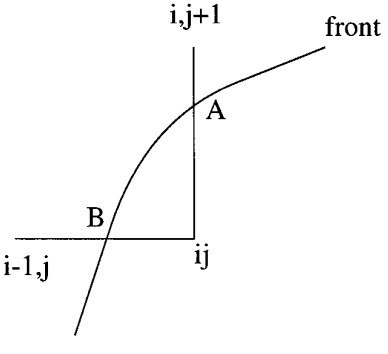

FIG. 3. Values of κ and V at points on the front (A and B) are interpolated from neighboring gridpoints.

TABLE I

Exact Solution: Moving Flat Interface: Convergence with Refining Both Grid Size and Timestep

Gridpts	Stepsize	Timestep	$\|T - T^n\|_{L_1}$	Conv. rate
80	0.025	1.00×10^{-4}	5.3657×10^{-7}	
160	0.0125	2.50×10^{-5}	1.3455×10^{-7}	1.9956
320	0.00625	6.25×10^{-6}	3.3664×10^{-8}	1.9988

Note. $N = 100, 400, 1600$ timesteps; final time $= 0.01$, $V = 1$.

discretization of Eqs. (10) and (19), we follow work done in [25] and use a simple TVD–Ruge–Kutta type time discretization. We use a semidiscrete, method of lines approach to solving the PDEs and we use a third-order Runge–Kutta scheme in time.

Thus we consider the equation

$$\phi_t = \mathscr{L}(\phi), \tag{22}$$

where $\mathscr{L}$ is the spatial operator of either Eq. (10) or (19). The time discretization of Eq. (22) is

$$\begin{aligned}
\phi^{(1)} &= \phi^{(0)} + \Delta t\, \hat{L}(\phi^{(0)}) \\
\phi^{(2)} &= \tfrac{3}{4}\phi^{(0)} + \tfrac{1}{4}\phi^{(1)} + \tfrac{1}{4}\Delta t\, \hat{L}(\phi^{(1)}) \\
\phi^{(3)} &= \tfrac{1}{3}\phi^{(0)} + \tfrac{2}{3}\phi^{(2)} + \tfrac{2}{3}\Delta t\, \hat{L}(\phi^{(2)}),
\end{aligned}$$

where $\hat{L}$ is the discrete approximation to $\mathscr{L}$ and should not be confused with the constant latent heat of solidification. Note that the timestep used in the above equations depends upon the particular partial differential equation we are solving. When we update ϕ by Eq. (10), we use the main timestep Δt. When we reinitialize ϕ by Eq. (19), we use a different timestep denoted by Δt_{reinit}.

4.4. *Discretization of Updating of the Level Set Function*

To discretize Eq. (10), we need to compute $F_{i,j}$, which in turn is computed from $u^k_{i,j}$, $k = 1 \cdots 4$. The following discrete equations are equally valid:

TABLE II

Exact Solution: Moving Flat Interface: Convergence Results from Refining Grid Size with Fixed Timestep

Gridpts	Stepsize	$\|T - T^n\|_{L_1}$	Conv. rate
80	0.025	1.3281×10^{-3}	
160	0.0125	2.6265×10^{-4}	2.3382
320	0.00625	3.6651×10^{-5}	2.8412
640	0.003125	3.7552×10^{-6}	3.2887

Note. $N = 400$ timesteps, $V = 1$, $dt = 0.00001$.

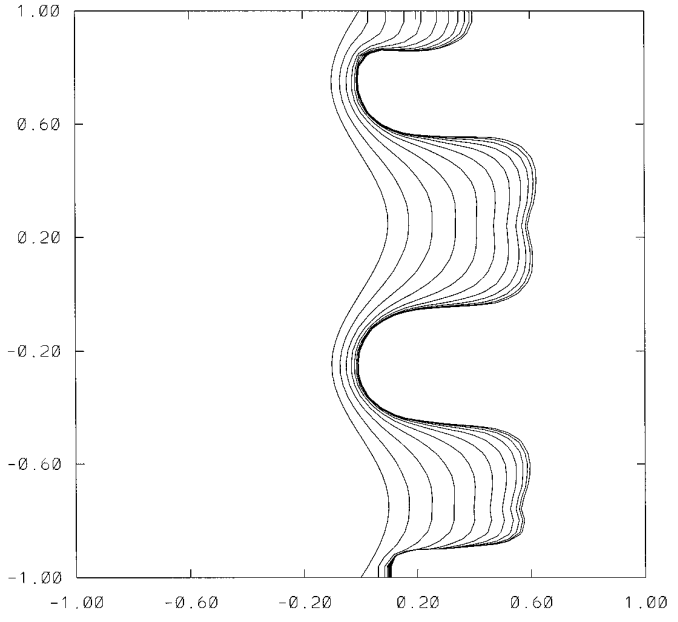

FIG. 4. Mullins–Sekerka instability: small perturbation to a flat interface will induce unstable dendritic growth.

$$F_{i,j} = u^1_{i,j}\left(\frac{\phi_x}{|\nabla\phi|}\right)_{i,j} + u^2_{i,j}\left(\frac{\phi_y}{|\nabla\phi|}\right)_{i,j}$$

$$F_{i,j} = u^3_{i,j}\left(\frac{\phi_\eta}{|\nabla\phi|}\right)_{i,j} + u^4_{i,j}\left(\frac{\phi_\zeta}{|\nabla\phi|}\right)_{i,j}.$$

So we use an average of the two expressions for $F_{i,j}$ to obtain

$$\begin{aligned}
F_{i,j} = \frac{1}{2}\Bigg(&u^1_{i,j}\left(\frac{\phi_x}{|\nabla\phi|}\right)_{i,j} + u^2_{i,j}\left(\frac{\phi_y}{|\nabla\phi|}\right)_{i,j} \\
&+ u^3_{i,j}\left(\frac{\phi_\eta}{|\nabla\phi|}\right)_{i,j} + u^4_{i,j}\left(\frac{\phi_\zeta}{|\nabla\phi|}\right)_{i,j}\Bigg).
\end{aligned} \tag{23}$$

Hence, $F_{i,j}|\nabla\phi| = \frac{1}{2}(u^1_{i,j}(\phi_x)_{i,j} + u^2_{i,j}(\phi_y)_{i,j} + u^3_{i,j}(\phi_\eta)_{i,j} + u^4_{i,j}(\phi_\zeta)_{i,j})$.

The spatial first derivatives of ϕ in the above relation are approximated by a second-order ENO scheme. From Eq. (10), we end up solving for the right-hand side of the equation

$$\phi_t = -\tfrac{1}{2}(u^1_{i,j}(\phi_x)_{i,j} + u^2_{i,j}(\phi_y)_{i,j} + u^3_{i,j}(\phi_\eta)_{i,j} + u^4_{i,j}(\phi_\zeta)_{i,j}).$$

Here ϕ is updated using a third-order Runge–Kutta scheme in time and a second-order ENO scheme in space.

For $\varepsilon_V \neq 0$, we compute $F_{i,j}$ explicitly since values of $F_{i,j}$ will be used later on to approximate the normal velocity at points on the front. The discrete approximation to $|\nabla\phi|$

TABLE III

Exact Solution: Growing Frank Spheres: Convergence Results from Refining Both Grid and Timestep

Gridsize	Stepsize	$\|T - T^n\|_{L_1}$	Timesteps	Conv. rate
80 × 80	0.2	4.3709×10^{-2}	3000	
160 × 160	0.1	1.4331×10^{-2}	12000	1.608
320 × 320	0.05	3.7938×10^{-3}	48000	1.917

Note. Final time = 1.12.

is computed using central difference approximations to ϕ_x, ϕ_y, ϕ_η, and ϕ_ζ. Hence,

$$|\nabla\phi|_{i,j} = \begin{cases} \sqrt{(\phi_x)_{i,j}^2 + (\phi_y)_{i,j}^2} & \text{or} \\ \sqrt{(\phi_\eta)_{i,j}^2 + (\phi_\zeta)_{i,j}^2}, \end{cases}$$

depending upon where the approximation to $|\nabla\phi|$ is being used. Finally, $F_{i,j}$ can be computed using Eq. (23) and the above discrete approximations to $|\nabla\phi|$.

4.5. *Discretization of the Reinitialization of ϕ*

In two dimensions, Eq. (19) can be rewritten as

$$\phi_t = S(\phi_0)(1 - \sqrt{\phi_x^2 + \phi_y^2}). \tag{24}$$

As shown in [27], one way of discretizing Eq. (24) is by Godunov's method

$$\phi_{i,j}^{N+1} = \phi_{i,j}^N - \Delta t S_\varepsilon(\phi_{i,j}^0) G(\phi_{i,j}^N), \tag{25}$$

where

$$a \equiv D_x^- \phi_{i,j} = (\phi_{i,j} - \phi_{i-1,j})/h$$
$$b \equiv D_x^+ \phi_{i,j} = (\phi_{i+1,j} - \phi_{i,j})/h$$
$$c \equiv D_y^- \phi_{i,j} = (\phi_{i,j} - \phi_{i,j-1})/h$$
$$d \equiv D_y^+ \phi_{i,j} = (\phi_{i,j+1} - \phi_{i,j})/h$$
$$S_\varepsilon(\phi)_{i,j} = \frac{\phi_{i,j}}{\sqrt{\phi_{i,j}^2 + \varepsilon^2}}$$

and

$$G(\phi)_{i,j} = \begin{cases} \sqrt{\max((a^+)^2, (b^-)^2) + \max((c^+)^2, (d^-)^2)} - 1, & \text{if } \phi_{i,j}^0 > 0, \\ \sqrt{\max((a^-)^2, (b^+)^2) + \max((c^-)^2, (d^+)^2)} - 1, & \text{if } \phi_{i,j}^0 < 0, \\ 0, & \text{otherwise.} \end{cases}$$

Equation (25) is a first-order, consistent, monotone scheme for solving Eq. (24). To achieve higher order accuracy in space, we replace the first-order backward and

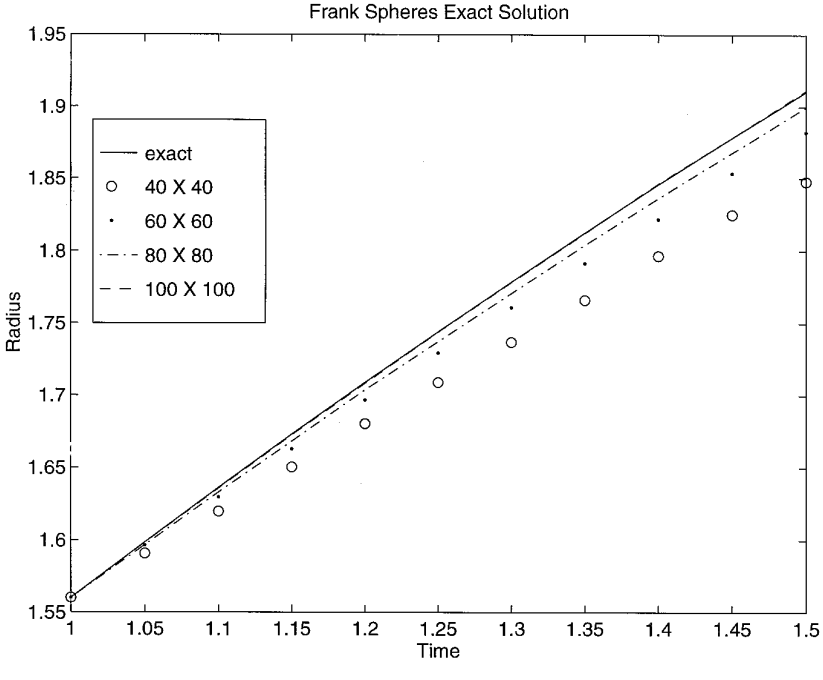

FIG. 5. For increasing grid sizes, the radii of the computed solution converges to that of the exact solution in the case of growing Frank spheres.

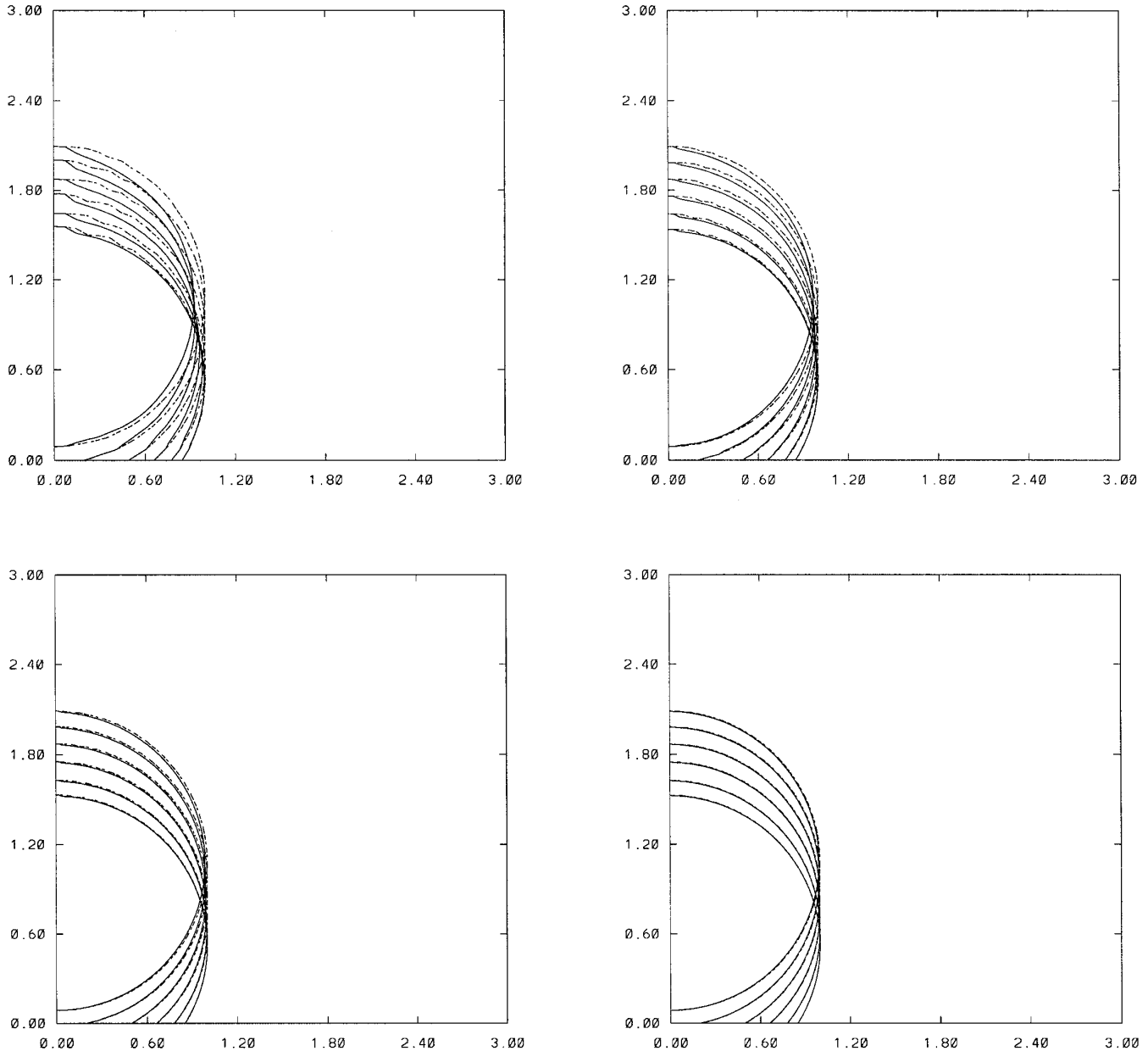

FIG. 6. The computed solution (solid line) converges to the exact solution (dotted line) for the case of an oscillating interface. Grid sizes used are: 40×40 (top left), 80×80 (top right), 160×160 (bottom left), and 320×320 (bottom right).

forward difference approximations to ϕ_x and ϕ_y by second-order ENO approximations. As detailed in Section 4.3, we use a method of lines approach to discretize Eq. (24) and our timestepping scheme is a third-order Runge–Kutta scheme. Thus, our way of discretizing Eq. (24) is TVD (total variation-diminishing).

In our computation of S_ε, we take $\varepsilon = 2h$ for smoothing purposes, which are needed when $\phi_{i,j}$ is close to zero. Since the timestep, Δt_{reinit}, taken is again not related to the main timestep Δt, we take $\Delta t_{\text{reinit}} = h/5$.

We iterate our scheme for Eq. (24) by a fixed number of iterations. Typically only three or four iterations are necessary in order for ϕ to be sufficiently evolved close enough to a distance function.

4.6. *Discretization of Temperature Update*

As mentioned in Section 3.4, T is updated by solving the heat equation for T in all of D. We use an implicit scheme to solve the heat equation in order to avoid any harsh timestep constraints.

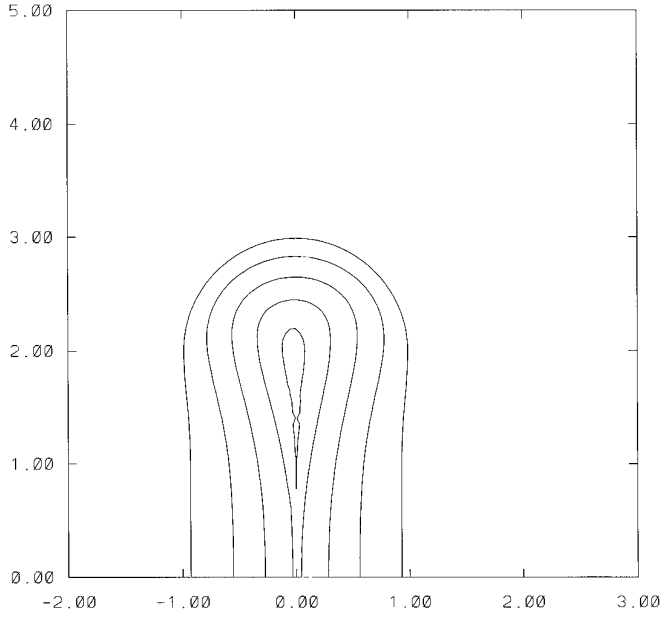

FIG. 7. Cusp formation.

Away from Γ, T is updated by the standard 5-point stencil scheme:

$$\frac{T_{i,j}^{n+1} - T_{i,j}^{n}}{\Delta t} = \frac{T_{i+1,j}^{n+1} - 2T_{i,j}^{n+1} + T_{i-1,j}^{n+1}}{h^2} + \frac{T_{i,j+1}^{n+1} - 2T_{i,j}^{n+1} + T_{i,j-1}^{n+1}}{h^2}. \tag{26}$$

With $\phi_{i,j}^{n+1}$, we check at every gridpoint to see whether or not a certain gridpoint borders the front in either the horizontal or vertical direction. For those points which do neighbor Γ, we compute the distance (horizontal and/or vertical) from those gridpoints to points on the front. Since $\phi_{i,j}^{n+1}$ is equal to a distance function from the front, these distances are quite easy to compute.

For example, suppose $\mathbf{x}_f \in \Gamma$ and $\mathbf{x}_f = (x_f, (j-1)h)$ for some integer j. Let us consider the two gridpoints $\mathbf{x}_{i,j}$ and $\mathbf{x}_{i+1,j}$ which border $\mathbf{x}_f$, i.e. where $x_i \le x_f \le x_{i+1}$. The distances

$$x_{i+1} - x_f = \left(\frac{\phi_{i+1,j}}{\phi_{i+1,j} - \phi_{i,j}}\right) h = r_1 h \tag{27}$$

$$x_f - x_i = -\left(\frac{\phi_{i,j}}{\phi_{i+1,j} - \phi_{i,j}}\right) h = r_2 h \tag{28}$$

are used when we construct interpolating polynomials P^{L} and P^{R}. (Note $r_1 + r_2 = 1$.) P^{L} and P^{R} are both functions of x only. We construct P^{L} from the values of h, r_2, $T(\mathbf{x}_f)$, $T_{i,j}$, $T_{i-1,j}$, $T_{i-2,j}$, etc. and P^{R} from the values of h, r_1, $T(\mathbf{x}_f)$, $T_{i+1,j}$, $T_{i+2,j}$, $T_{i+3,j}$, etc.

With the coefficients of the interpolated polynomials computed beforehand, we can differentiate each polynomial twice to obtain approximations to the double derivatives with respect to x. Thus at $\mathbf{x}_{i,j}$, $T_{xx} \approx P_{xx}^{\mathrm{L}}$ and at $\mathbf{x}_{i+1,j}$, $T_{xx} \approx P_{xx}^{\mathrm{R}}$. Note that since we are using an implicit scheme, P^{L} and P^{R} are always constructed in the abstract sense from values of $T_{i,j}^{n+1}$.

Taking into account both the gridpoints away from and near Γ, the general form of the discrete equation we solve is

$$C_{i,j} T_{i,j}^{n+1} = T_{i,j}^{n} + f_{i,j}(T_{i+1,j}^{n+1}, T_{i-1,j}^{n+1}, T_{i,j+1}^{n+1}, T_{i,j-1}^{n+1}, \ldots). \tag{29}$$

We have found that a simple way of solving this nonlinear discrete equation is by the Gauss–Seidel method. Our stopping criterion is

$$\sum_{i,j=1}^{M+1} (T_{i,j}^{N+1} - T_{i,j}^{N}) < \mathrm{tol}, \quad \text{where tol} = 1.0 \times 10^{-12}.$$

4.7. *Discretization of Curvature*

From the Gibbs–Thomson relation, Eq. (4), we see that $T(\mathbf{x}_f)$ is dependent upon $\kappa(\mathbf{x}_f)$. Thus, when we construct the interpolating polynomials as outlined in Section 4.6, approximations to $\kappa(\mathbf{x}_f)$ are needed.

The expression we use for curvature, κ, is given by Eq. (21). In this formula for κ, ϕ_x, ϕ_y, ϕ_{xx}, and ϕ_{yy} are all discretized by central differencing. The mixed derivative term ϕ_{xy} is discretized by

$$(\phi_{xy})_{i,j} = \frac{\phi_{i+1,j+1} - \phi_{i+1,j-1} - \phi_{i-1,j+1} + \phi_{i-1,j-1}}{4h^2}. \tag{30}$$

We do not calculate the curvature at every gridpoint since it is only necessary to compute κ at gridpoints bordering the front. If $\varepsilon_C \neq 0$, then the value of κ on Γ is interpolated from the value of κ at neighboring gridpoints. Similarly, for $\varepsilon_V \neq 0$, the discrete velocity field $F_{i,j}$ is used to approximate $V(\mathbf{x}_f)$ for use in the interpolating polynomials.

In Fig. 3, the front intersects the grid at two places, denoted by A and B. Let us denote the curvature and normal velocity at these two points by κ_A, κ_B, V_A, and V_B. We approximate κ_A and κ_B from values of ϕ at neighboring gridpoints, and V_A and V_B are approximated from values of F at neighboring gridpoints. From Eq. (4), $T(A) = -\varepsilon_C \kappa_A - \varepsilon_V V_A$ and $T(B) = -\varepsilon_C \kappa_B - \varepsilon_V V_B$. At the gridpoint (x_i, y_{j+1}), the value of $T(A)$ is used in the approximation of T_{yy}. At (x_{i-1}, y_j), $T(B)$ is used in the approximation to T_{xx}. Finally, at (x_i, y_j), T_{xx} is approximated using $T(B)$ and T_{yy} is approximated using $T(A)$.

4.8. *Discretization of Anisotropic Terms*

In the supercooled liquid case, anisotropy affects the shape of the growing crystal, causing it to grow along pre-

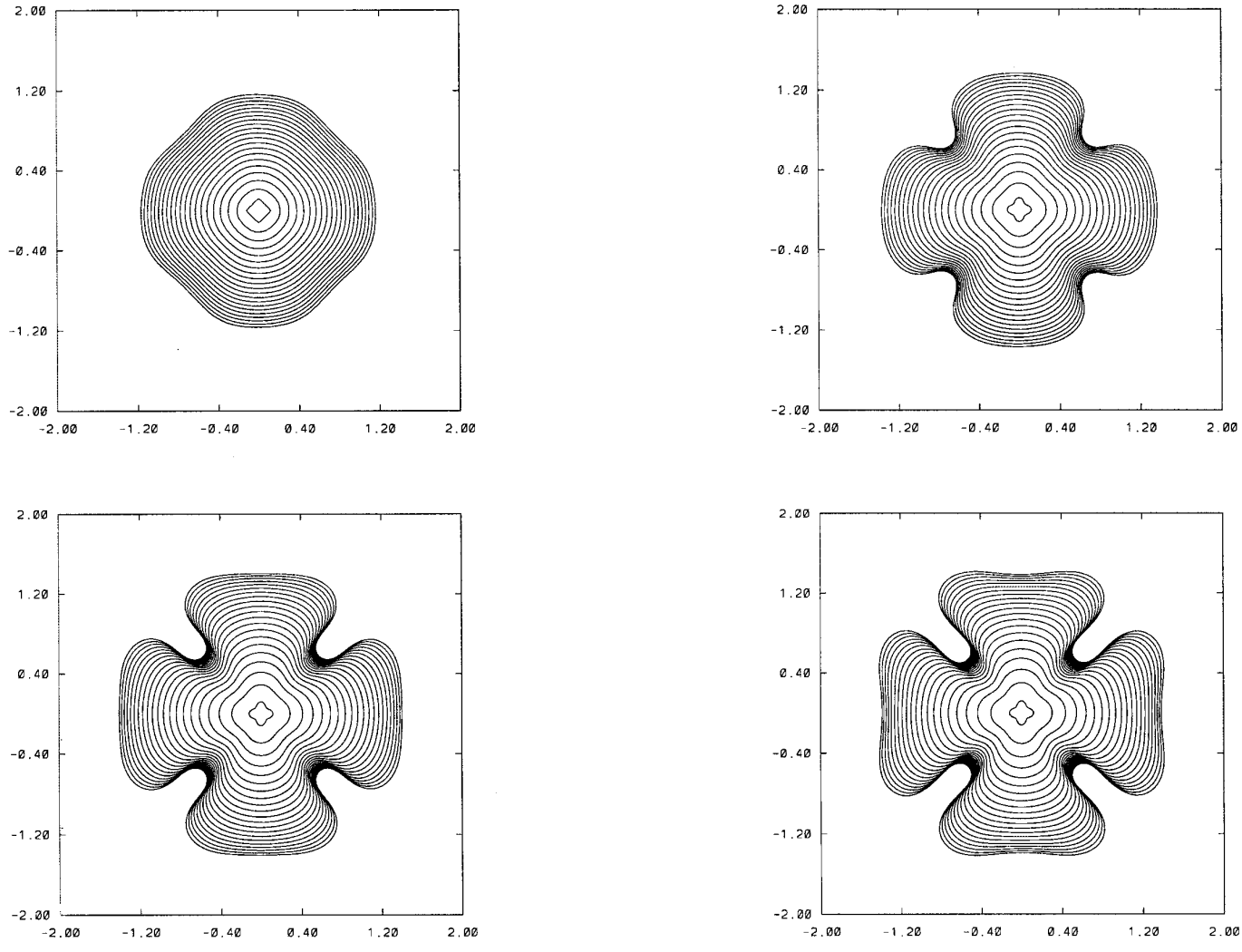

FIG. 8. Convergence study: growth histories for 4 grid resolutions. The grid sizes used are: 100 × 100 (top left), 200 × 200 (top right), 300 × 300 (bottom left), and 400 × 400 (bottom right).

ferred axial directions. Many examples of anisotropic growth occur in nature; hence, it is important for an algorithm to be able to accurately model anisotropic growth.

In Eqs. (5) and (6), we see that the surface tension and molecular kinetic terms in the Gibbs–Thomson relation can be anisotropic. That is, rather than being constant, we can set ε_C and ε_V to be dependent upon the normal vector $\mathbf{n}$.

When we discretize Eqs. (5) and (6), we need to approximate the angle θ that the normal vector $\mathbf{n}$ makes with the x-axis. For $\mathbf{x}_1 = (1, 0)$,

$$\mathbf{x}_1 \cdot \mathbf{n} = |\mathbf{x}_1|\,|\mathbf{n}| \cos(\theta) = \cos(\theta). \tag{31}$$

But Eq. (11) implies that

$$\mathbf{x}_1 \cdot \mathbf{n} = \phi_x/|\nabla\phi|. \tag{32}$$

Thus we compute θ by taking the inverse cosine of $\phi_x/|\nabla\phi|$ at gridpoints neighboring the front. Then we interpolate the value of the angle that a given point on the front makes with the x-axis. The discretization of $\phi_x/|\nabla\phi|$ is computed from central difference approximations to ϕ_x and ϕ_y.

5. NUMERICAL RESULTS

In this section, we present the results found when we applied our algorithm to solving Stefan problems and to modeling unstable crystal growth. For exact solutions to the Stefan problem, we applied our method to see how accurate it was and how fast it converged to given exact solutions. Then, we adjusted parameters in our algorithm to mimic the conditions corresponding to dendritic solidi-

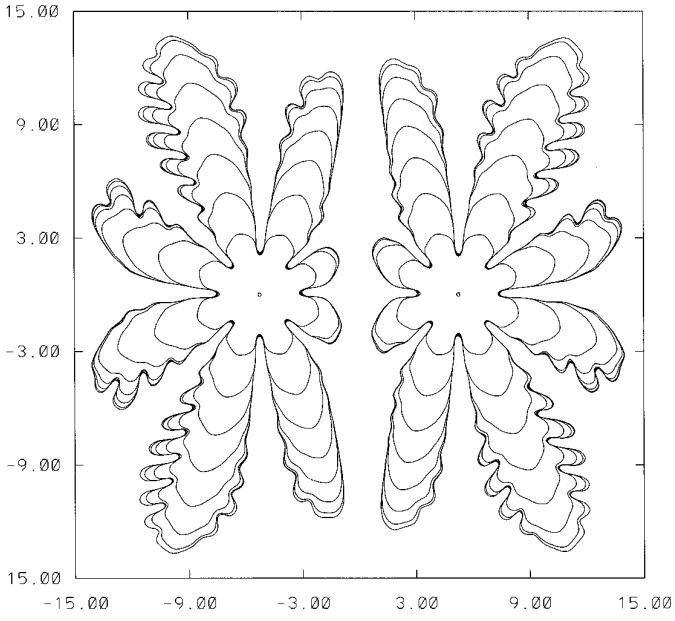

FIG. 9. Multiple enclosed fronts: $T = 0$ on Γ; hence there is a numerical regularization effect from the level set method. The computation above was performed with $\Delta x = 0.1$, $\Delta t = 0.01$, $T_\infty = -0.5$ and time levels shown are in increments of 3.

fication. More specifically, we tested our algorithm to see whether or not it was able to accurately simulate the effects of surface tension, kinetic effects, and anisotropy on a growing crystal.

5.1. *Exact Solutions*

We tested our algorithm for solving Stefan problems on some exact solutions. Our exact solutions include a moving flat interface, a growing spherical interface and an oscillating circular front. Moreover, we tested the algorithm to see if it could simulate Mullins–Sekerka instability and also to see whether it could correctly evolve a front into a cusp.

5.1.1. *Moving Planar Interface.* For a steadily advancing planar surface, $\Omega(t) = \{\mathbf{x} \in D : x < Vt\}$, where the normal velocity V is constant, an exact solution to the Stefan problem is given by

$$T(\mathbf{x}, t) = \begin{cases} -1 + e^{-V(x - Vt)}, & x > Vt, \\ 0, & x \le Vt. \end{cases} \tag{33}$$

The interface in Eq. (33) is parameterized by

$$\Gamma(t) = \{x = VT, y = s\}, \quad s \in \Re. \tag{34}$$

In two dimensions, Γ is just a line moving with constant speed V.

Applying our method to the solution above and measuring the error between the exact and computed solutions, we have determined that the method is second order in space for the one-dimensional case. We measure the error in the L_1 norm. Our results are shown in Tables I and II.

5.1.2. *Mullins–Sekerka Instability.* By the Mullins–Sekerka analysis, it has been shown (see [16] for details) that for the classical Stefan problem ($T = T_m$ on Γ), a perturbation to a flat interface will grow arbitrarily large for positive V. By perturbing the interface so that

$$x = Vt + \varepsilon e^{\lambda t} \sin(ky) \tag{35}$$

one can obtain an expression for the perturbed temperature field that is $O(\varepsilon)$ and of the form:

$$T(x, y, t) = T_o(x, y, t) + \varepsilon e^{\lambda t} \sin(ky) \begin{cases} f_1(k, V), & x > Vt, \\ f_2(k, V), & x < Vt \end{cases} \tag{36}$$

(see [1, 26] for more details). For small times, our method

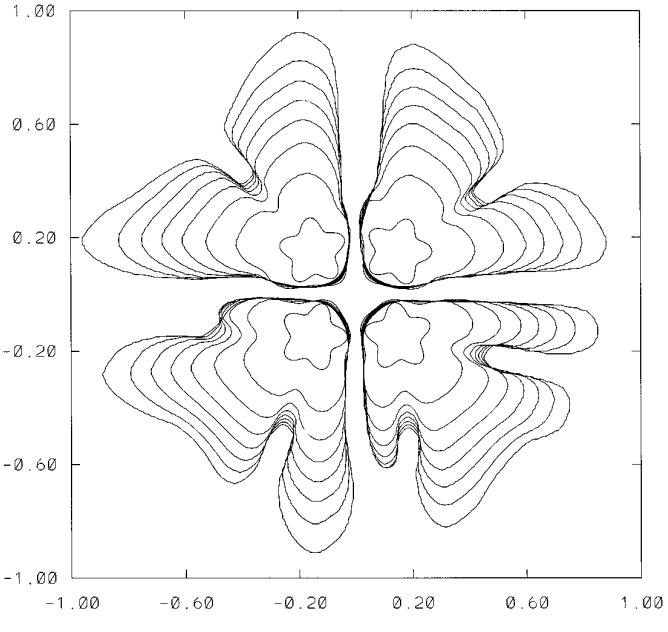

FIG. 10. Multiple enclosed fronts: four seeds growing with $\varepsilon_C = 0.001$ and $\varepsilon_V = 0.001$. The computation above was performed with $\Delta x = 0.01$, $\Delta t = 0.0005$, $T_\infty = -1.0$ and time levels shown are in increments of 0.025.

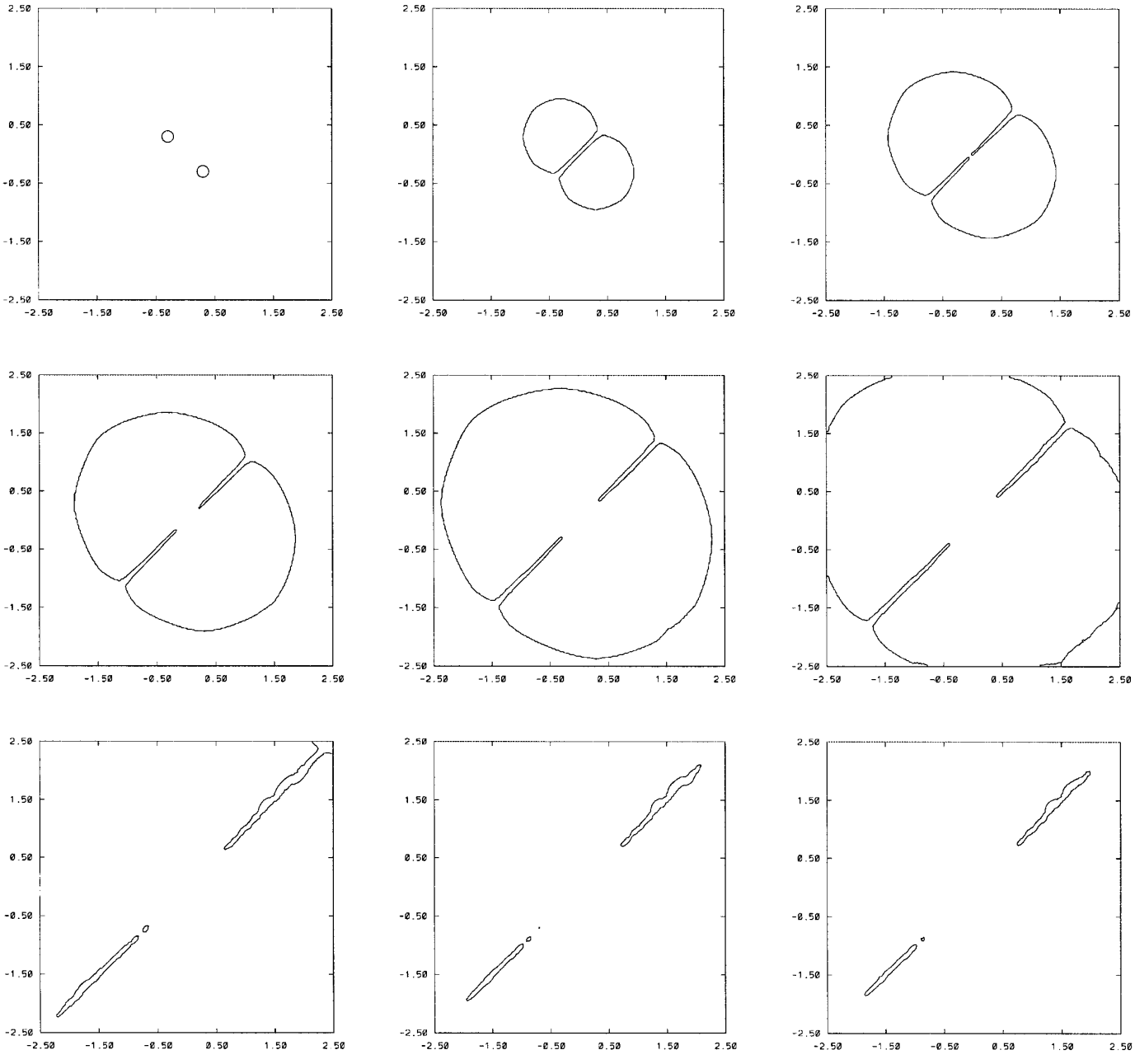

FIG. 11. Merging fronts: case of two solid spheres in supercooled liquid. Time levels shown from top left to bottom right: $t = 0.0$, 0.01, 0.02, 0.03, 0.04, 0.05, 0.08, 0.1, 0.11.

converges near the interface to the solution given by the perturbed temperature field.

Figure 4 is a plot generated from the Mullins–Sekerka initial data above (Eqs. (35), (36)). Various positions of the front in time are shown, illustrating the fact that small perturbations to a flat interface will grow arbitrarily large. The figure was generated with no surface tension or kinetic effects and with insulated boundary conditions. The plot was generated on a 50×50 grid with $\Delta t = 0.001$, up to a final time of 1. Level curves are shown at times $0, 0.1, \cdots, 0.9, 1$ with T set to 0 on the interface.

5.1.3. *Growing Frank Spheres.* For the problem in two dimensions, there is an exact solution for the classical Stefan problem called the growing Frank spheres solution, with formulas found in [1]. Here the solid region is a cylinder of radius $R = St^{1/2}$ and the temperature field $T(r, t)$ is given by

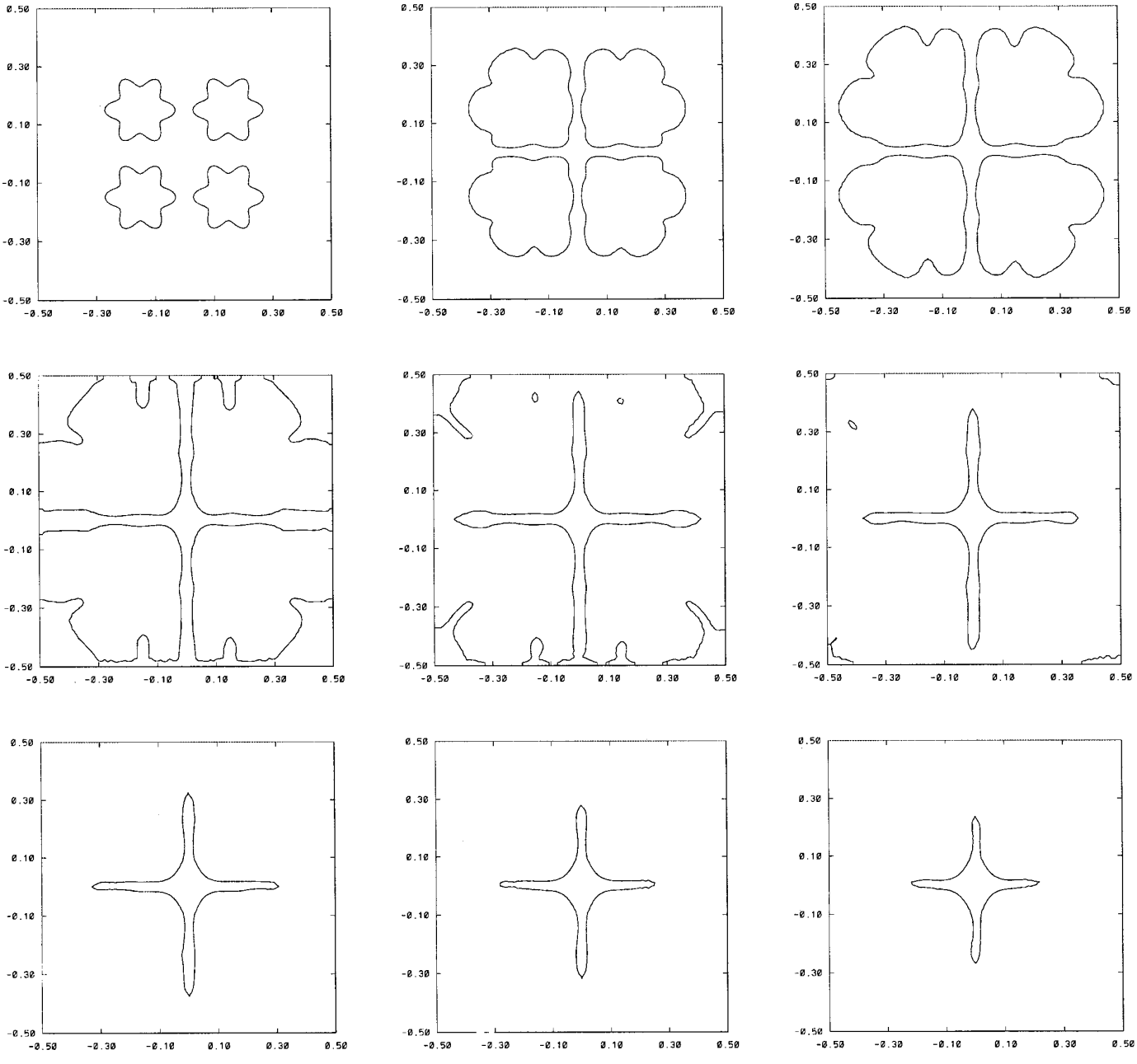

FIG. 12. Merging fronts: case of four seeds with fixed boundary conditions. Time levels shown from top left to bottom right: $t = 0.0$, 0.004, 0.008, 0.012, 0.016, 0.02, 0.024, 0.028, 0.032.

$$T(r,t) = T(s) = \begin{cases} T_\infty \left(1 - \dfrac{F(s)}{F(S)}\right), & s > S, \\ 0, & s < S, \end{cases} \tag{37}$$

where $r = \sqrt{x^2 + y^2}$, $s = r/t^{1/2}$, and T_∞ is a given undercooling. The function $F(s)$ is a similarity solution of the heat equation:

$$F(s) = E_1(\tfrac{1}{4}s^2). \tag{38}$$

Some numerical results using the Frank spheres solutions are shown in Table III. For our comparison with the Frank spheres solutions, we took $T_\infty = -0.5$, the spatial domain to be 16×16, and homogeneous Neumann (insulated) boundary conditions. In Fig. 5, we plot the radius

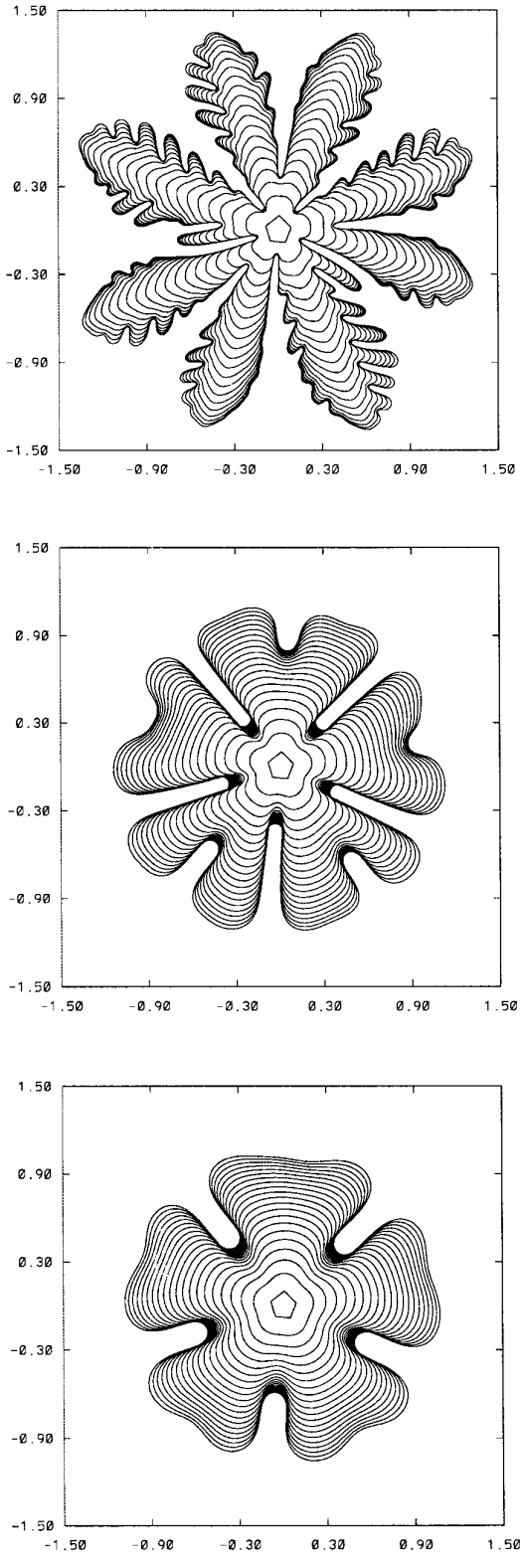

FIG. 13. Effect of varying isotropic surface tension; the initial seed is an irregular pentagon. For all 3 plots, $\varepsilon_V = 0$. For the top plot, $T = 0$ on Γ. For the middle and bottom plots, $T = -\varepsilon_C\kappa$ on Γ with $\varepsilon_C = 0.0005$ (middle) and $\varepsilon_C = 0.001$ (bottom).

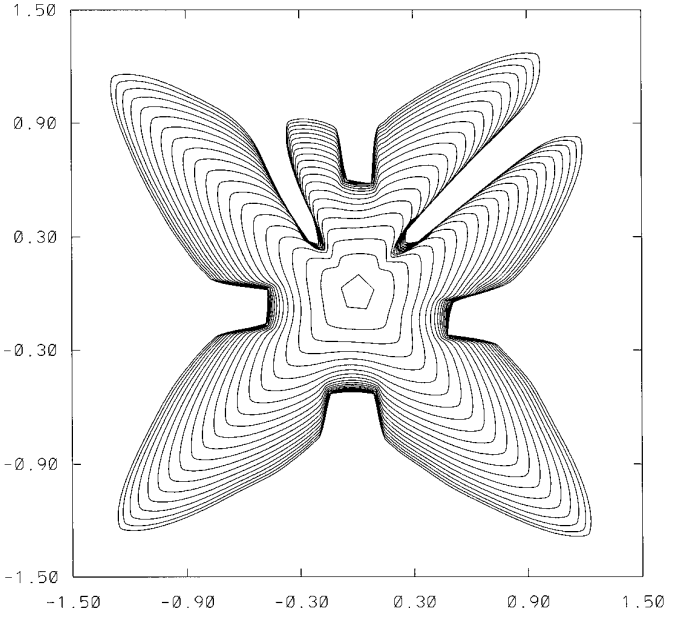

FIG. 14. Effect of anisotropic surface tension; the initial seed is an irregular pentagon. On Γ, $T = -\varepsilon_C(\mathbf{n})\kappa$ and $\varepsilon_{\bar{C}} = 0.001$, $\varepsilon_V = 0$.

error between the exact and computed solutions for the Frank spheres data for increasingly finer grids in space.

5.1.4. *Oscillating Spherical Fronts.* Another exact solution to the Stefan problem was constructed in a paper by Nochetto, Verdi, and Paolini [17]. In their paper, they constructed an exact solution to the Stefan problem that consists of an interface that is an oscillating sphere. Figure 6 is a plot of the exact (dotted) and computed (solid) interface curves for various times and grid sizes. We plot the exact versus the computed solutions for $\Delta t = 0.002$, and for 40×40, 80×80, 160×160, and 320×320 grid sizes. The time intervals shown are at 0.05, 0.1, 0.2, 0.3, 0.4, and 0.5.

5.1.5. *Cusp Formation.* Also, in [17], it was shown that certain prescribed initial data for the Stefan problem should lead to the formation of a cusp. For the adaptive finite element method presented in [17], this initial data is a good test to see how the numerical mushy zone around

TABLE IV

Convergence of Anisotropic Tip Speeds

Gridsize	$\varepsilon_C = 0.0005$	$\varepsilon_C = 0.001$	$\varepsilon_C = 0.002$
75×75	37.17	38.94	35.62
100×100	37.17	42.48	38.05
125×125	50.44	49.56	43.36
150×150	53.10	51.33	44.25

TABLE V

Convergence of Anisotropic Tip Radii

Gridsize	$\varepsilon_C = 0.0005$	$\varepsilon_C = 0.001$	$\varepsilon_C = 0.002$
75×75	3.197×10^{-2}	3.685×10^{-2}	3.147×10^{-2}
100×100	1.497×10^{-2}	2.771×10^{-2}	2.080×10^{-2}
125×125	2.103×10^{-2}	1.886×10^{-2}	1.503×10^{-2}
150×150	1.605×10^{-2}	1.446×10^{-2}	1.302×10^{-2}

the cusp is resolved. In our method, cusps can be resolved easily by the level set function. Using the initial data in section 7.3 of [17], we did indeed find that the interface between the two phases eventually evolved into a cusp, as shown in Fig. 7. This figure was generated on a 100×100 grid with $h = 0.05$, $\Delta t = 0.002$. The time levels shown are 0.02, 0.22, 0.42, 0.62, 0.82.

5.2. *Experimental Results with Unstable Solidification*

We now come to some of the results obtained by simulating the growth of a solid into an undercooled liquid. In all of these experiments, we initially set $T = T_\infty < 0$ everywhere in D except for a small region or area, where we set $T = 0$. This is to simulate the conditions of supercooling, where a small frozen seed of material is placed in a surrounding region of undercooled liquid. Unless otherwise stated, the boundary conditions taken in these experiments are insulated, i.e., homogeneous Neumann boundary conditions.

In most of our computations, we incorporated the Gibbs–Thomson relation by setting the temperature equal to some combination of curvature and interfacial velocity at each timestep. However, in a few cases (Figs. 9 and 13), we set $T = 0$ on the front for all time. Physically, of course, the case when $T = 0$ on Γ should be impossible to resolve numerically because the problem itself is an ill-posed one. The fact that we obtain pictures that do not "blow up" illustrates an interesting feature of level set methods. Level set methods have both a topological and a curvature regularization built in to them. Thus when you apply a level set method, unstable interfaces are regularized automatically. This regularization effect on ill-posed problems is discussed in more depth in [5], where the level set method is applied to the Cauchy–Riemann equations.

5.2.1. *Refining the Grid.* For small values of ε_C and ε_V, we tested our method to see if given an initial interfacial shape, the evolution of the interface over time remained the same for different grid resolutions. As illustrated in Fig. 8, for increasing grid sizes, the plots of the zero level sets of ϕ do appear to converge to a similar shape over time. The initial interfacial shape is given explicitly by

$$x(s) = (R + P\cos(8\pi s))\cos(2\pi s)$$

$$y(s) = (R + P\cos(8\pi s))\sin(2\pi s),$$

where $R = 0.1$ and $P = 0.02$.

We take the spatial domain to be $[-2, 2] \times [-2, 2]$, and the grid sizes to be 100×100, 200×200, 300×300, and 400×400. For each plot in Fig. 8, we set $\Delta t = 0.0005$, $\varepsilon_C = 0.002$, $\varepsilon_V = 0.002$, and $T_\infty = -0.5$. Time levels shown are in increments of 0.04 up to a final time of 0.8. The convergence of these plots under grid refinement compares favorably with the results generated by the front-tracking method in [7].

5.2.2. *Multiple Fronts and Topology Changes.* Our level set method easily handles complicated interfacial geometries. Figures 9 and 10 demonstrate that the method can numerically simulate cases when there are multiple frozen seeds surrounded by undercooled liquid. Figure 9 is a plot of the evolution of two initially square shaped seeds as they grow towards one another. As mentioned above, since T is set equal to 0 on Γ, Fig. 9 does not really represent the actual solution to the model equations; rather it supports the fact that the level set method introduces a numerical regularization for ill-posed problems. Figure 9 was generated on a 300×300 grid with fixed outer boundary conditions, the undercooled temperature T_∞ set to -0.5 and the domain $D = [-15, 15]$. Along with $\Delta x = 0.1$, we took $\Delta t = 0.01$ and ran the computation up to a final time of 24. The time levels shown in Fig. 9 are in increments of 3. In Fig. 10, we have four identically shaped seeds growing in close proximity to one another. In contrast to Fig. 9, we simulate the effects of isotropic surface tension and kinetic effects by taking $\varepsilon_C = 0.001$ and $\varepsilon_V = 0.001$. At the boundary, we fix $T = T_\infty = -1$. Figure 10 was generated on a 200×200 grid, with $\Delta t = 0.0005$, $D = [-1, 1] \times [-1, 1]$, and $L = 1$. The time levels shown are $t = 0, 0.025, 0.05, 0.075, ..., 0.175$. No anisotropy was added via the Gibbs–Thomson relation in either Fig. 9 or Fig. 10. But in Fig. 9, it is clear there is some grid-induced anisotropy. In part, we attribute this to the coarse spatial stepsize taken ($\Delta x = 0.1$) because in Fig. 10, when a finer mesh size is used ($\Delta x = 0.01$), there are no observable artificial anisotropic effects.

Another topological change the level set method handles easily is the merging of fronts. Unlike most front tracking methods (e.g., [7]), no special conditions need to be imposed on our method when the boundaries between two solid regions grow arbitrarily close. Figures 11 and 12 are two cases of fronts merging. In Fig. 11, we see the evolution of two circular seeds as they grow into one another. This figure was plotted on a 200×200 grid, with $D = [-2.5, 2.5] \times [-2.5, 2.5]$, $\varepsilon_C = 0.001$, $\varepsilon_V = 0.001$, $T_\infty = -1$, $L = 0.075$, and $\Delta t = 0.0001$. With the temperature at the boundary walls fixed, we see the seeds growing toward the walls, as well as toward one another. Eventually, two small re-

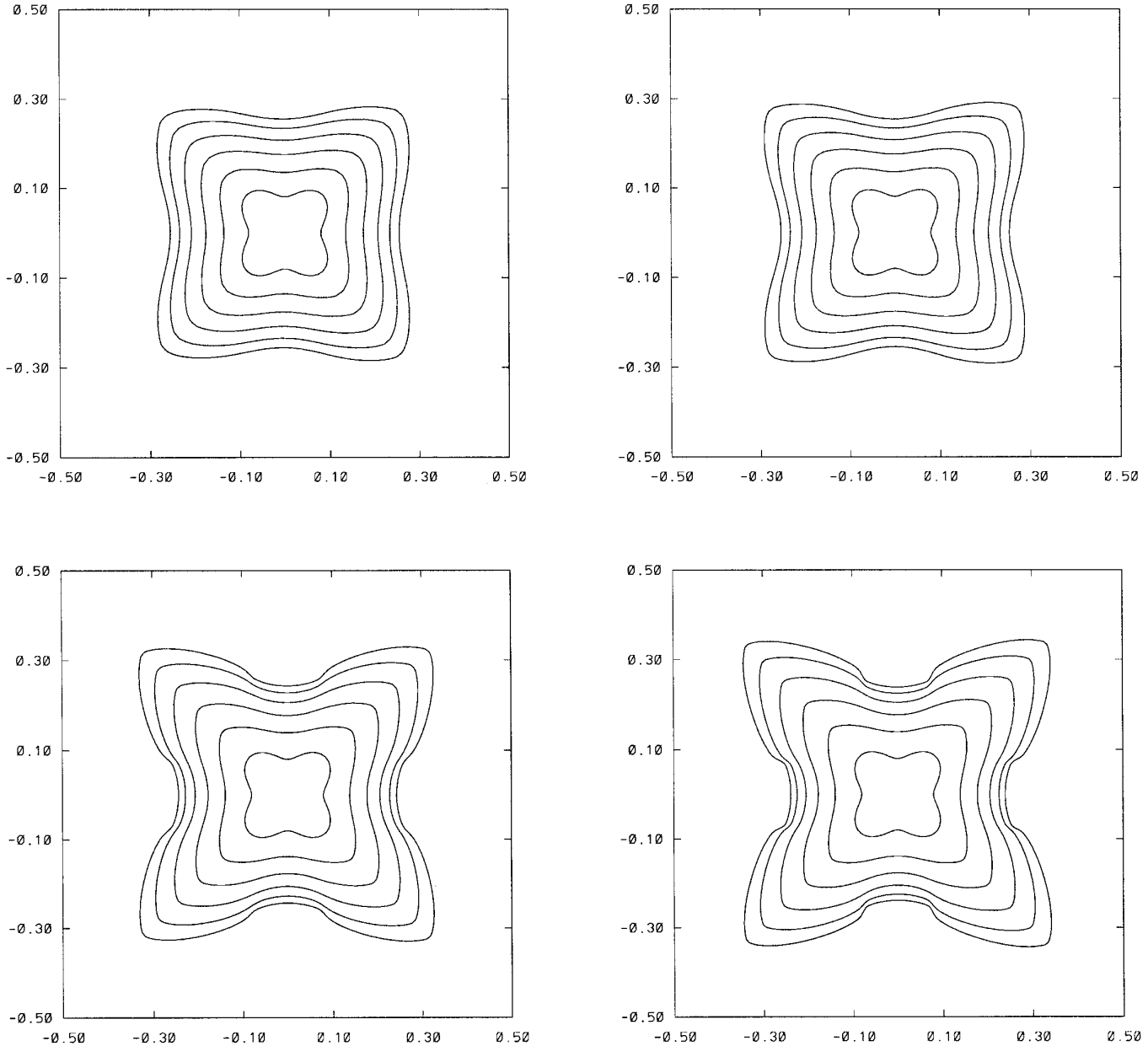

FIG. 15. Convergence of anisotropic tip speeds with $\varepsilon_{\bar{C}} = 0.001$. Grids are: 75×75 (top left), 100×100 (top right), 125×125 (bottom left), 150×150 (bottom right); final time $= 0.05$.

gions of entrapped liquid form. The nine pictures shown in Fig. 11 are the evolution of the crystal at times $t = 0.0$, 0.01, 0.02, 0.03, 0.04, 0.05, 0.08, 0.1, 0.11. Similarly, in Fig. 12 the evolution of four six-lobed crystalline shapes is shown as they first merge into the box walls and then into one another. In Fig. 12, the times shown are $t = 0.0$, 0.004, 0.008, 0.012, 0.016, 0.02, 0.024, 0.028, and 0.032. Each plot in Fig. 12 was generated on a 100×100 grid, with $D = [-0.5, 0.5] \times [-0.5, 0.5]$, $\varepsilon_C = 0.001$, $\varepsilon_V = 0.001$, $T_\infty = -1$, and $L = 0.5$. We keep T fixed at T_∞ on the boundary walls.

5.2.3. *Varying the Surface Tension Coefficient* ε_C. The three plots in Fig. 13 were all generated by the same initial conditions and with $\varepsilon_V = 0$. What was varied among these plots was the amount of imposed isotropic surface tension. The top plot in Fig. 13 shows the evolution of the frozen seed when $T = 0$ on Γ. Again, as was the case in Fig. 9, this plot is evidence that the level set method introduces numerical regularization for the ill-posed problem, as well as a discernable artificial anisotropic effect. Interestingly enough, this artificial grid induced anisotropy decreases

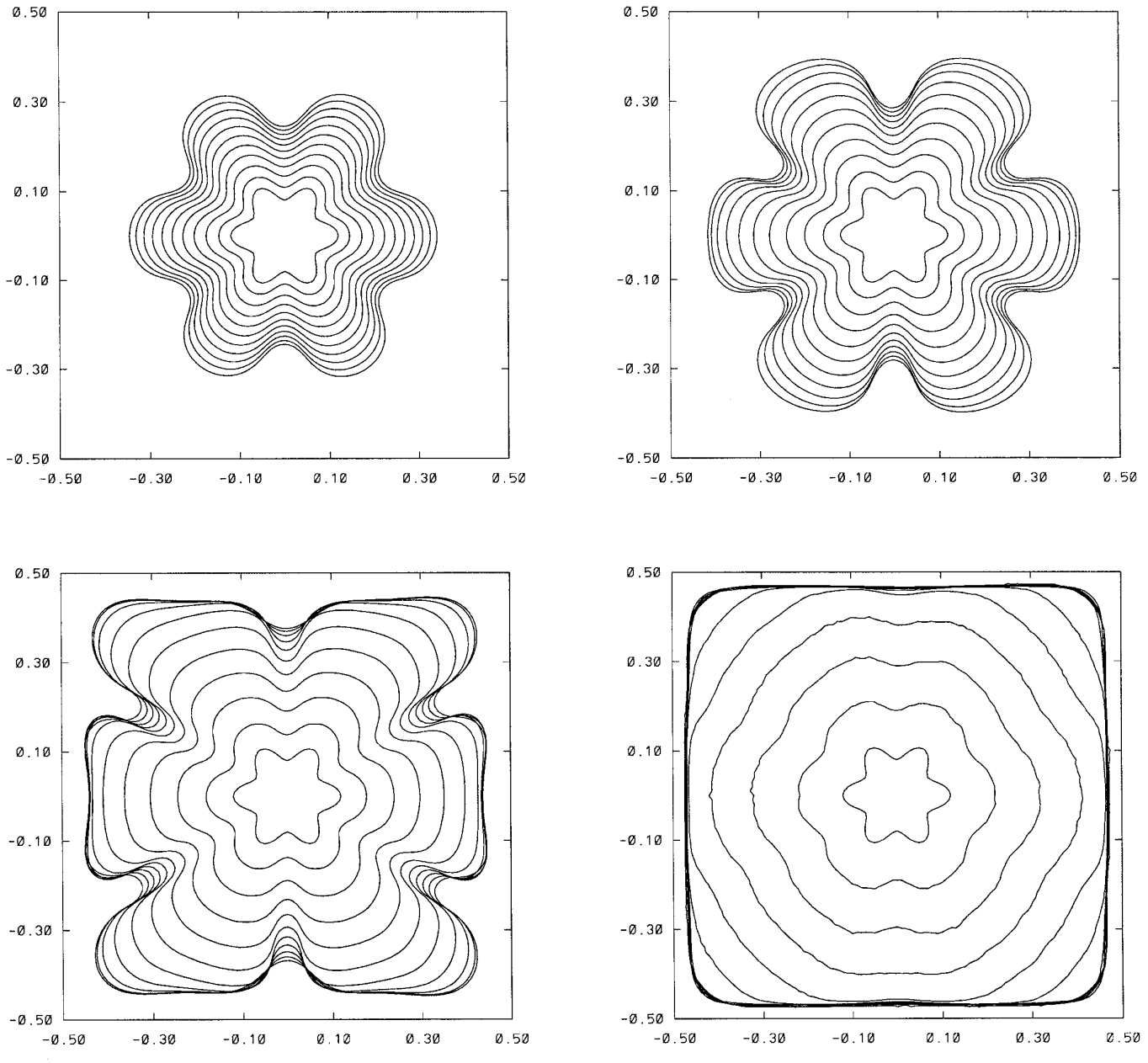

FIG. 16. Effect of varying the latent heat of solidification: L = 1.0 (top left), 0.75 (top right), 0.5 (bottom left), and 0.25 (bottom right).

when we increase the value of the isotropic surface tension coefficient. This can be seen in the bottom and middle plots of Fig. 13, where $T = -\varepsilon_c \kappa$ on Γ, with $\varepsilon_C = 0.0005$ in the middle plot and $\varepsilon_C = 0.001$ in the bottom plot. Hence, in the lower two plots, it is more apparent how perturbations in the initial interfacial shape (i.e., the five corners of the irregular pentagon) influence the evolution of the front to its final shape. We conclude that the imposed isotropic surface tension has a stabilizing effect on the unstable problem, decreasing the grid induced anisotropy effect and causing the front to evolve by an isotropic process that eventually leads to tip splitting. All three plots in Fig. 13 were generated on a 300×300 grid with $h = 0.01$, $\Delta t = 0.0005$, and $N = 800$ timesteps. The time levels shown are in increments of 0.02. We set $T_\infty = -0.5$ and ran up to a final time of 0.4.

5.2.4. *Results with Anisotropy*. Crystalline anisotropy will cause a material to grow along preferred lines or axes. In Fig. 14, we add anisotropy to the curvature term follow-

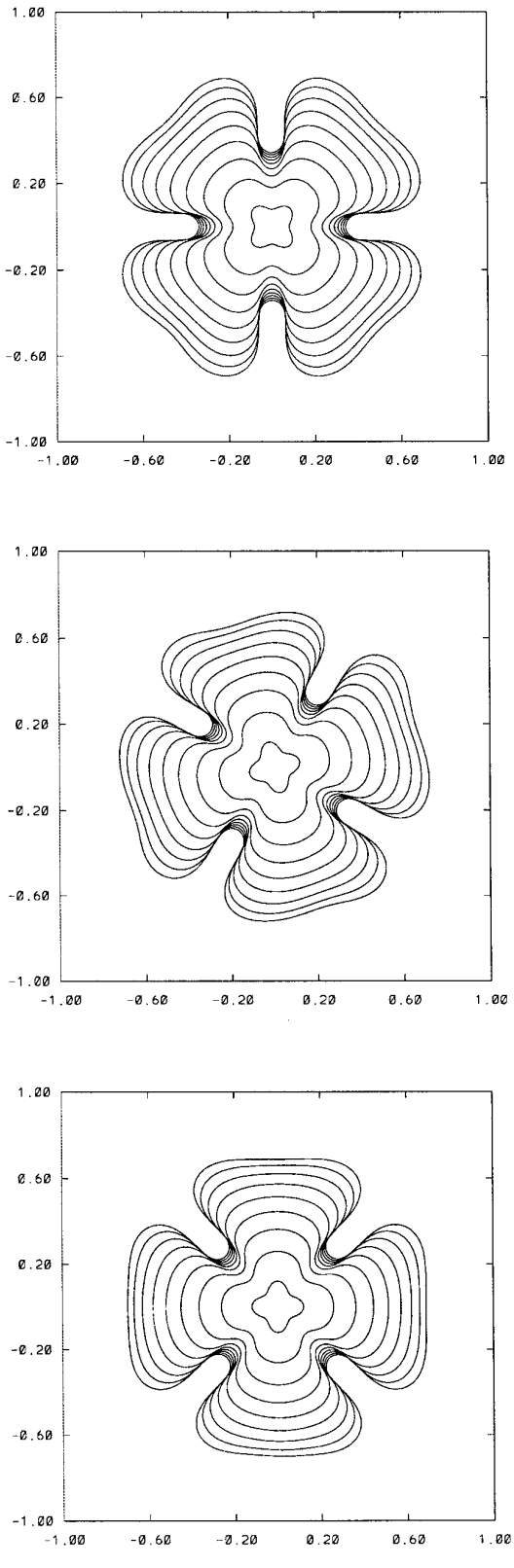

FIG. 17. Grid orientation effects with isotropic surface tension. With $\varepsilon_C = 0.001$, grid orientation effects are minimal. From top to bottom, the initial data is rotated 45°, 35°, and 0°.

ing the expression used in [1]. We take the anisotropic curvature term to be of the form

$$\varepsilon_C(\mathbf{n}) = \bar{\varepsilon_C}(\tfrac{8}{3}\sin^4(\tfrac{1}{2}m(\theta - \theta_o))). \tag{39}$$

Here m is the mode number which we set equal to 4 and θ_o is the phase angle which we set equal to $\pi/4$. For $\bar{\varepsilon_C} > 0$, we expect the crystal to grow along the four diagonal axes rather than retain its initial perturbed shape. In Fig. 14, we see that the imposed fourfold anisotropy causes the dendrite to favor growth along the diagonal directions. From the top corner of the pentagonal seed, tip splitting occurs early on. But, eventually, the split tip begins growing towards the diagonal corners. Figure 14 was generated by the same initial data and conditions used for Fig. 13 (see Section 5.2.3 for details).

5.2.5. *Convergence of Anisotropic Tip Speeds.* In [6], Ivantsov showed that for a paraboloid shaped dendrite, there is a relationship given by VR = constant, between the dendritic tip speed and radius. Ever since then, researchers have been trying to determine the velocity of crystals using Ivantsov's relation coupled with predictions from either marginal stability analysis or microscopic solvability theory. From the latter, there is the postulation that the unique dendritic operating state depends upon values of anisotropic surface tension. Many numerical methods [9, 21, 28] for dendritic solidification have been used to perform quantitative tests comparing the measured speeds and radii of dendritic tips with predictions from dendritic growth theory. From a numerical standpoint, it is logical to require that as the grid size is refined, the measures of the tip speed and radius should converge; otherwise, it may be hard to justify using such a numerical method to perform quantitative analyses.

With our method, we did not perform any quantitative comparisons with dendritic growth theory. Rather, we just checked to see whether or not the tip speeds and radii converged as we refined the grid. In Table IV, we display the measured tip speeds for four grid sizes and three values of $\bar{\varepsilon_C}$: 0.0005, 0.001, and 0.002. In Table V, we show the corresponding values of the radii of the anisotropic tips. In Fig. 15, we present the plots generated by the same initial data used in Tables IV and V when the magnitude of the surface tension coefficient is equal to 0.001 and as the grid sizes vary from 75×75 to 150×150. Each plot in Fig. 15 was generated with $\varepsilon_V = 0$ and $T_\infty = -0.5$. Time increments are shown in multiples of 0.01 up to a final time of 0.05. The domain taken in each plot is $[-0.5, 0.5]$. Anisotropy is included by setting ε_C equal to the expression in Eq. (39). We set $\theta_0 = \pi/4$ so that the tips will grow along the diagonal axes. We measured the tip speeds by taking the average of the growth of each of the four tips. We measured the tip radii by calculating the curvature at the

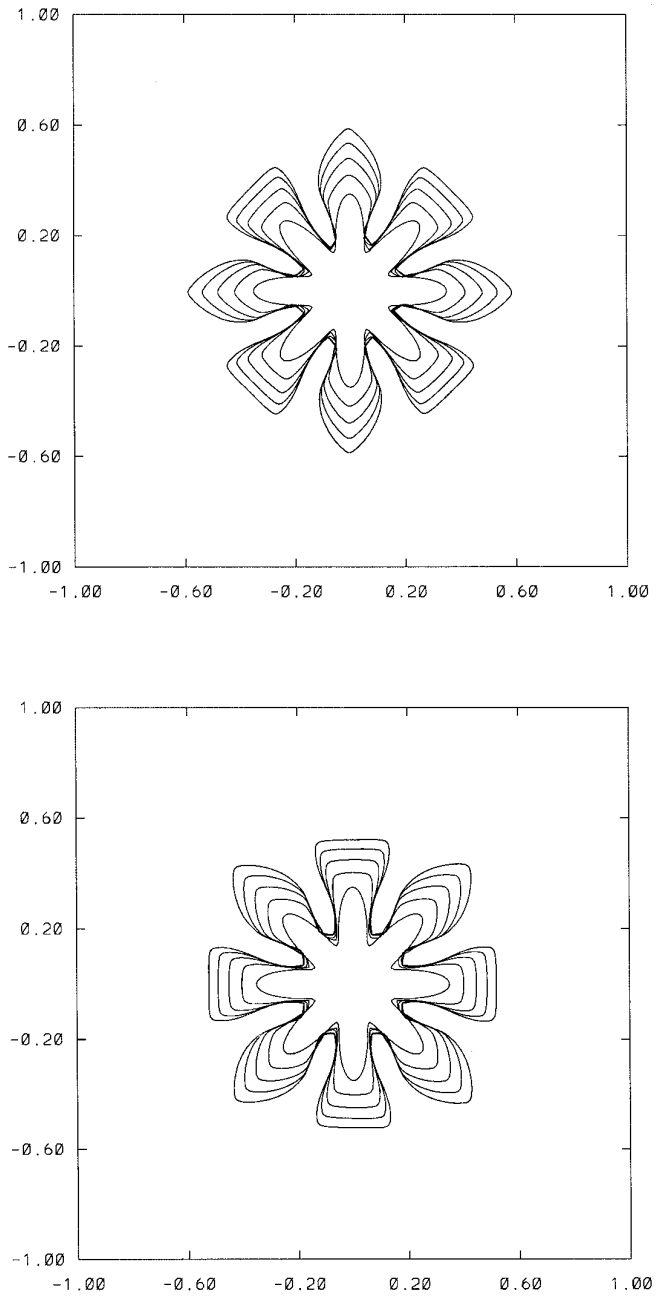

FIG. 18. Grid orientation effects with anisotropy: the anisotropy is fourfold with phase angle θ_o equal to 0° (top) and $\pi/4$ (bottom).

tip and taking its reciprocal. From Tables IV and V and Fig. 15, it is clear that, as the grid is refined, the measured tip speeds and radii converge. We observe that as ε_C^- increases, we see a much better convergence of the tip speeds, which is to be expected since linear stability analysis proves that surface tension will have a stabilizing effect on the unstable problem.

5.2.6. *Varying the Latent Heat of Solidification.* Figure 16 is a study of the effect of changing the latent heat of solidification. As Sethian and Strain point out in [24], setting $L = 0$ reduces the motion of the crystal to pure geometry. Decreasing the latent heat of solidification has a smoothing effect on the crystalline shape. By the results of linear instability analysis, increasing L translates into increasing the range of unstable wavelengths. In Fig. 16, we see that the computational results of increasing L agree with our theoretical expectations. The four plots in Fig. 16 were run on a 100×100 grid up to a final time of 0.05. We took $\Delta t = 0.0005$ and plotted the contour levels at time increments of 0.005. Here the undercooled temperature T_∞ was set to -0.5, $\varepsilon_C = 0.001$, and $\varepsilon_V = 0.001$. The domain $D = [-0.5, 0.5] \times [-0.5, 0.5]$ and $h = 0.01$. We only considered the isotropic case. The latent heats of solidification in the four plots are taken to be 1.0, 0.75, 0.5, and 0.25.

5.2.7. *Grid Orientation Effects.* As detailed in Section 3.2, we compute the normal velocity components in four different coordinate directions in order to reduce grid orientation effects. Previously, we had only computed the normal velocity components in the Cartesian coordinate directions. What we found was that the same initial data rotated at different angles did not evolve into the same final shape rotated at the original angles. In searching for ways to reduce these grid effects, we found that by including the normal velocity components from the diagonalized Cartesian coordinate system, we were able to decrease grid effects considerably.

In Fig. 17, we took the same initial shape and rotated it by 45° and 35°. Our computations were done on a 200×200 grid, with $h = 0.01$, $\Delta t = 0.0005$, $T_\infty = -0.5$, and $N = 400$ timesteps. The final time was 0.2 and in each of the three plots shown in Fig. 17, the time levels shown are in multiples of 0.025. We took $\varepsilon_C = 0.001$ and $\varepsilon_V = 0$, and we only considered the isotropic case. As can be seen in this figure, the final interfacial shapes correspond well to the rotated angles of the initial data. Our results compare favorably with those of [7].

We considered that another test of grid orientation effects was to vary the angle of anisotropy. Figure 18 shows the effect of varying the phase angle θ_o on a crystal growing with fourfold anisotropic surface tension and anisotropic kinetic effects. The initial seed is a synmmetric shape with eight smooth bumps. In Eqs. (5) and (6), the coefficient k_A controls the number of folds in the anisotropy. Here we set $k_A = 4$ and take $\theta_o = 0$ in the top plot and $\theta_o = \pi/4$ in the bottom plot.

The number of anisotropic folds and the direction of the preferred growth axes determines the crystalline shape. When $\theta_o = 0^o$, dendritic growth is favored along the horizontal and vertical axes. In the top plot of Fig. 18, we see that the horizontal and vertical tips sharpen, while the diagonal tips flatten out. When $\theta_o = \pi/4$, the exact opposite

occurs, with the horizontal and vertical tips flattening out, while the diagonal tips sharpen. As to be expected, the top and bottom plots in Fig. 18 match up when either plot is rotated 45°. Both plots in Fig. 18 were generated using a mesh size of 200 × 200, and by setting $h = 0.01$, $\varepsilon_{\bar{C}} = 0.001$, $\varepsilon_{\bar{V}} = 0.001$, $\Delta t = 0.0005$, and the domain $D = [-1, 1] \times [-1, 1]$. The final time taken is 0.04 and all contour levels shown are in increments of $t = 0.01$.

6. CONCLUSIONS

In conclusion, we have presented a new numerical method for solving the Stefan problem and the related model problem of unstable dendritic solidification. Our method is a level set method which implicitly evolves the sharp interface between two phases of a material. As such, our method is able to simulate complicated interfacial shapes involving merging, side branching and dendritic fingering. Physically, our model includes such effects as crystalline anisotropy, curvature, and interface kinetics.

Under grid refinement, we have found that this method converges to certain exact solutions of the Stefan problem. For dendritic structures, our method also converges when appropriate amounts of surface tension and kinetic effects are included. Moreover, we found that this method exhibits minimal grid effects when initial shapes are rotated. When we add anisotropy, our method correctly simulates growth along prescribed axes of symmetry. Also, we have found that with this method, the measured tip speeds of growing anisotropic dendrites converge under grid refinement.

In the future, we would like to apply this method to make more quantitative comparisons with results from dendritic growth theory. Furthermore, since this level set method is relatively easy to implement and computationally inexpensive, we would like to extend it to simulating the growth of three-dimensional dendrites. It would be interesting to compare our results with other methods that have already simulated three-dimensional crystalline structures [8, 10, 23].

ACKNOWLEDGMENTS

The authors thank R. Sekerka for many helpful discussions concerning the physics of this problem.

REFERENCES

1. R. Almgren, Variational algorithms and pattern formation in dendritic solidification, *J. Comput. Phys.* **106,** 337 (1993).
2. K. Brattkus and D. I. Meiron, Numerical simulations of unsteady crystal growth, *SIAM J. Appl. Math.* **52,** 1303 (1992).
3. G. Caginalp and E. A. Socolovsky, Computation of sharp phase boundaries by spreading: the planar and spherically symmetric cases, *J. Comput. Phys.* **95,** 85 (1991).
4. G. Fix, Phase field models for free boundary problems, in *Free Boundary Problems: Theory and Applications, Vol. II*, edited by A. Fasano and M. Primicerio, pp. 580–600 (Piman, Boston, 1983).
5. E. Harabetian and S. Osher, ''Regularization of Ill-Posed Problems Via the Level Set Approach,'' UCLA CAM Report 95-41, 1995. [*SIAM J. Appl. Math.*, to appear]
6. G. P. Ivantsov, Temperature field around spherical, cylindrical and acircular crystal growing in a supercooled melt, *Dokl. Akad. Nauk SSSR* **58,** 567 (1947).
7. D. Juric and G. Tryggvason, A front tracking method for dendritic solidification, *J. Comput. Phys.* **123,** 127 (1996).
8. A. Karma and W. J. Rappel, Numerical simulation of three-dimensional dendritic growth, preprint. [*Phys. Rev. Lett.*, to appear].
9. A. Karma and W. J. Rappel, Phase-field method for computationally efficient modeling of solidification with arbitrary interface kinetics, *Phys. Rev. E* **53,** 3017 (1996).
10. R. Kobayashi, Modeling and numerical simulations of dendritic crystal growth, *Physica D* **63,** 410 (1993).
11. J. S. Langer, Instabilities and pattern formation in crystal growth, *Rev. Mod. Phys.* **52,** 1 (1980).
12. J. S. Langer, Lectures in the theory of pattern formation, in *Chance and Matter*, edited by J. Souletie, J. Vannimenus, and R. Stora, pp. 629–711 (North Holland, Amsterdam, 1987).
13. J. S. Langer and H. Muller-Krumbhaar, Theory of dendritic growth, *Acta. Metall.* **26,** 1681 (1978).
14. A. Meirmanov, *The Stefan Problem*, (DeGruyter, Berlin, 1992), pp. 10–30.
15. B. Merriman, J. K. Bence, and S. J. Osher, Motion of multiple junctions: A level set approach, *J. Comput. Phys.* **112,** 334 (1994).
16. W. W. Mullins and R. F. Sekerka, Stability of a planar interface during solidification of a dilute binary alloy, *J. Appl. Phys.* **35,** 444 (1964).
17. R. H. Nochetto, M. Paolini, and C. Verdi, An adaptive finite element method for two-phase Stefan problems in two space dimensions. Part II: Implementation and numerical experiments, *SIAM J. Sci. Stat. Comput.* **12,** 1207 (1991).
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. P. Pelce, *Dynamics of Curved Fronts* (Academic Press, San Diego, 1988), pp. 1–102.
20. O. Penrose and P. C. Fife, Thermodynamically consistent models of phase-field type for the kinetics of phase transitions, *Physica D* **43,** 44 (1990).
21. A. R. Roosen and J. E. Taylor, Modeling crystal growth in a diffusion field using fully faceted interfaces *J. Comput. Phys.* **114,** 113 (1994).
22. M. E. Rose, An enthalpy scheme for Stefan problems in several dimensions, *App. Numerical Math.* **12,** 229 (1993).
23. A. Schmidt, Computation of three dimensional dendrites with finite elements, *J. Comput. Phys.* **125,** 293 (1996).
24. J. Sethian and J. Strain, Crystal growth and dendritic solidification, *J. Comput. Phys.* **98,** 231 (1992).
25. C. W. Shu and S. Osher, Efficient implementation of essentially non-oscillatory shock-capturing schemes, II *J. Comput. Phys.* **77**(2), 439 (1988).
26. J. Strain, Linear stability of planar solidification fronts, *Physica D* **30,** 297 (1988).
27. M. Sussman, P. Smereka, and S. Osher, A level set approach for computing solutions to incompressible two-phase flow, *J. Comput. Phys.* **114,** 146 (1994).
28. S-L. Wang and R. F. Sekerka, Algorithms for phase field computation of the dendritic operating state at large supercoolings, *L. Comput. Phys.* **127,** 110 (1996).
29. J. Warren, How does a metal freeze? A phase-field model of alloy solidification, *Comput. Sci. Eng.* **2**(2), 38 (1995).