Journal of Scientific Computing, Vol. 19, Nos. 1–3, December 2003 (© 2003)

Semi-Implicit Level Set Methods for Curvature and Surface Diffusion Motion

Peter Smereka[1]

Received April 30, 2002; accepted (in revised form) on September 10, 2002

In this paper we introduce semi-implicit methods for evolving interfaces by mean curvature flow and surface diffusion using level set methods.

KEY WORDS: Curvature flow; surface diffusion; level set methods.

1. INTRODUCTION

The level set method, developed by Osher and Sethian [13] has had large impact on computational methods for interface motion and is now being used by engineers, physicists and mathematicians for a wide variety of problems ranging from two-fluid flows to epitaxial growth; see, for example, the recent reviews by Osher and Fedkiw [12], Sethian [16], and Sethian and Smereka [17].

In the level set method, the interface, Γ, is represented implicitly as the zero level set of a continuous function which we will denote as u; therefore we write

$$\Gamma = \{\mathbf{x} \mid u(\mathbf{x}, t) = 0\}.$$

If the normal speed of the interface is v_n then the time evolution of u is given by

$$\frac{\partial u}{\partial t} + v_n |\nabla u| = 0. \tag{1}$$

If we wish to consider the motion of interface whose normal speed is unity then (1) becomes

$$\frac{\partial u}{\partial t} + |\nabla u| = 0. \tag{2}$$

Dedicated to Stan Osher on the occasion of his 60th birthday.

[1] Department of Mathematics, University of Michigan, Ann Arbor, MI 48109. E-mail: psmereka@umich.edu

0885-7474/03/1200-0439/0 © 2003 Plenum Publishing Corporation

Equation (2) is a Hamilton–Jacobi equation. Osher and Sethian [13] outline the importance of upwind methods for Hamilton–Jacobi equations in level set applications.

Let us now consider the case when the normal velocity is $-\kappa$ where κ is the mean curvature. We may easily write the mean curvature in terms of u as follows; one has $\kappa = \nabla_s \cdot \mathbf{n}$ where $\nabla_s \cdot$ is the surface divergence and $\mathbf{n}$ is the outward drawn normal. Since $\nabla_s = \nabla - \mathbf{n}\partial_n$ and $\mathbf{n}\partial_n \cdot \mathbf{n} = 0$ then $\kappa = \nabla \cdot \mathbf{n}$ where $\nabla \cdot$ is the usual divergence operator. Furthermore the normal vector to a level curve of u is

$$\mathbf{n} = \frac{\nabla u}{|\nabla u|}; \tag{3}$$

hence

$$\kappa = \nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right). \tag{4}$$

Therefore, the level set equation for motion by mean curvature is:

$$\frac{\partial u}{\partial t} - \nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right) |\nabla u| = 0. \tag{5}$$

The main difficulty when solving this equation numerically is that it is stiff. This equation has the same numerical issues as the heat equation. The natural way of dealing with stiff problems is to use implicit methods; this is straightforward for the heat equation (in simple geometries) since it is linear. Unfortunately (5) is nonlinear, indicating that it would not be straightforward to implement an implicit method. On the other hand, if an explicit method is used then small time steps are needed to maintain stability. One way to circumvent this difficulty is to use the diffusion generated motion by mean curvature method developed by Merriman, Bence, and Osher [11]. However this approach has other difficulties which have been outlined by Ruuth [15] and he has developed a method using adaptive mesh refinement to improve the diffusion generated motion algorithm. In Section 2 we will outline a semi-implicit method for computing mean curvature flows which is stable for large time steps.

1.1. Motion by Surface Diffusion

Surface diffusion is an important physical effect that is present in many models of film growth; see, for example, Mullins [9, 10], Srolovitz *et al.* [18], Li *et al.* [8], and Adalsteinsson and Sethian [1]. In surface diffusion, atoms diffuse from areas of high mean curvature to low mean curvature. Therefore, surfaces of constant mean curvature are steady solutions to motion by surface diffusion. The normal velocity of an interface moving under surface diffusion is $\Delta_s \kappa$ where Δ_s is the surface Laplacian. This can by written in terms of the level set function as follows; first we write the surface Laplacian as

$$\Delta_s = \nabla_s \cdot \nabla_s \qquad \text{where} \quad \nabla_s = \nabla - \mathbf{n}\partial_n$$

with $\mathbf{n}$ given above and $\partial_n = \mathbf{n} \cdot \nabla$. Since both κ and $\mathbf{n}$ are easily expressed in terms of u then one can use the above formula to express $\Delta_s \kappa$ in terms of u. The equation of motion of the level set function for surface diffusion is

$$\frac{\partial u}{\partial t} + \Delta_s \kappa \, |\nabla u| = 0. \tag{6}$$

Level set methods for motion by surface diffusion have been developed by Chopp and Sethian [4], Khenner *et al.* [7] and Tasdizen *et al.* [20]. One of the fundamental difficulties associated with the numerical solution of surface diffusion problems is that they are very stiff. This is true no matter which method is used (front tracking, phase field, or finite element). In this respect it is analogous to solving $u_t = -u_{xxxx}$. If one uses an explicit method then the stability condition requires that $k < Ch^4$; this is very severe. Here k is the time step and h is the mesh size. Since the problem is nonlinear it is not easy to use implicit methods. In the work of Chopp and Sethian [4] and Khenner *et al.* [7] explicit methods are used and the authors use very small time steps. Tasdizen *et al.* [20] also use explicit methods but they introduce a novel approach, using a two step process, first computing the evolution of the surface normal and then construction of the surface. Isotropic surface diffusion is a special case of this method. This is applied to image processing.

Level set methods for this problem also suffer from an additional difficulty. In the level set approach one also solves for u not only near the interface but in all of the computational domain. As pointed out by Chopp and Sethian [4], this equation does not have a maximum principle, consequently an embedded curve may not remain so. In particular, new interfaces may spontaneously nucleate. Chopp and Sethian [4] have successfully ameliorated this problem by using narrow band level set methods and thereby only computing near the interface. The resulting method was quite successful in many aspects and they were able to compute several types of surface diffusion problems. In Section 3 we will present a new algorithm for computing motion by surface diffusion which is stable and easy to implement. This algorithm is based on an idea developed in Section 2 which we will now discuss.

2. CURVATURE FLOWS

As pointed out above, the mean curvature can easily be computed using the level set function. Our algorithm for curvature flow is based the formula,

$$|\nabla u| \, \kappa = \Delta u - N(u) \tag{7}$$

where

$$N(u) = \frac{\nabla u}{|\nabla u|} \cdot \nabla(|\nabla u|) = \partial_n \, |\nabla u|$$

which follows directly from (4). Therefore, the equations of motion for flow by mean curvature can now be written as:

$$\frac{\partial u}{\partial t} = \Delta u - N(u). \tag{8}$$

We observe that $N(u)$ is a nonlinear second order operator. In addition we see that if u is the signed distance function then $|\nabla u| = 1$ and $N(u) = 0$. Therefore we conclude that if u is parameterized to be close to distance function then $N(u)$ will be small. Based on this observation we discretize (7) in time as follows:

$$u^{n+1} = u^n + k\,\Delta u^{n+1} - kN(u^n) \tag{9}$$

where k is the time step. This method is first order accurate in time. The hope is that by treating the linear term implicitly this will stabilize explicit term even though $N(u)$ is a second order operator. This discretization is first order operator splitting where one first takes one step of forward Euler on the nonlinear term followed by one step of backward Euler on the linear term. In our computation, we find that

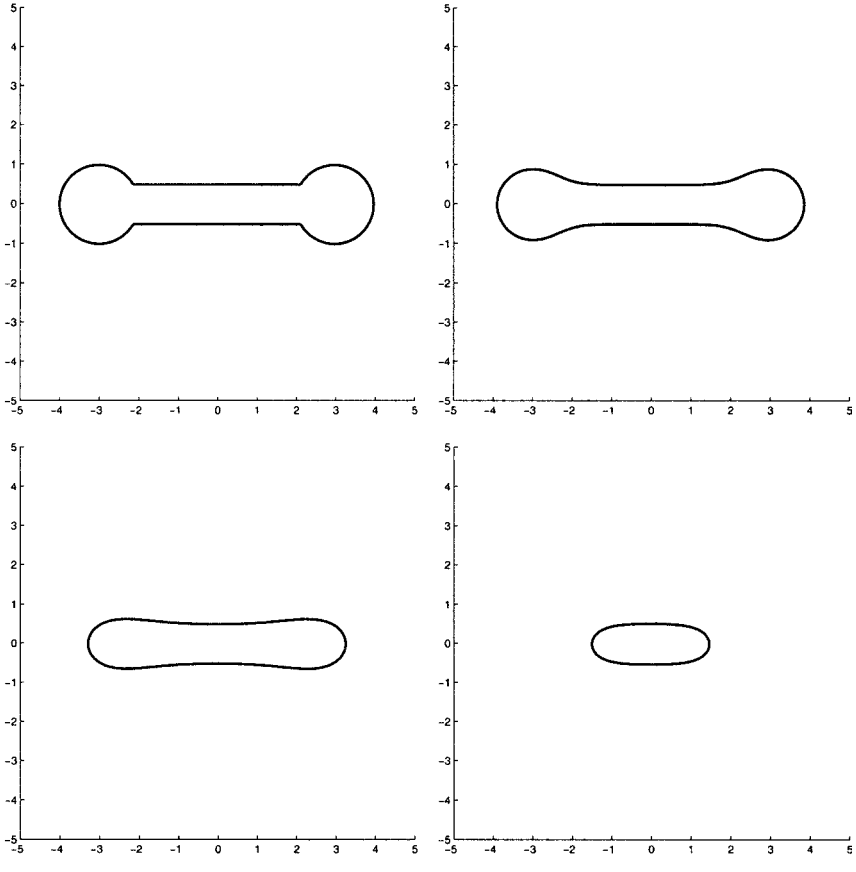

Fig. 1. Motion by mean curvature in two spatial dimensions. From the top left corner to the bottom right corner the times are $t = 0$, 0.01, 0.5, and 1.2. The time step is 0.01 and the mesh size is 0.0390625. The computation is done on a 256×256 grid.

the method appears to be stable even though we never reinitialize the level set function to be a distance function.

Equation (9) is spatially discretized by expressing Δu and $N(u)$ explicitly in terms of first and second order partial derivatives and then discretizing using center differences. For example, in two space dimensions we first write $N(u)$ and Δu as

$$N(u) = \frac{u_x^2 u_{xx} + 2u_x u_y u_{xy} + u_y^2 u_{yy}}{u_x^2 + u_y^2} \qquad \text{and} \qquad \Delta u = u_{xx} + u_{yy}$$

and then discretize with following center differences

$$u_x = \frac{u_{i+1,j} - u_{i-1,j}}{2h},$$

$$u_{xx} = \frac{u_{i-1,j} - 2u_{i,j} + u_{i+1,j}}{h^2},$$

and

$$u_{xy} = \frac{u_{i-1,j-1} + u_{i+1,j+1} - u_{i-1,j+1} - u_{i+1,j-1}}{4h^2}.$$

Other derivatives are dealt with in a similar fashion. Finally the denominator of $N(u)$ is replaced by $u_x^2 + u_y^2 + \varepsilon$. In our computations we use $\varepsilon = 10^{-12}$.

Our computations are done in a square in two dimensions and a cube in three dimension with periodic boundary conditions. In order to implement the scheme given by (9) we must invert the operator $I - k\Delta_h$; this is done using a FFT. Here Δ_h

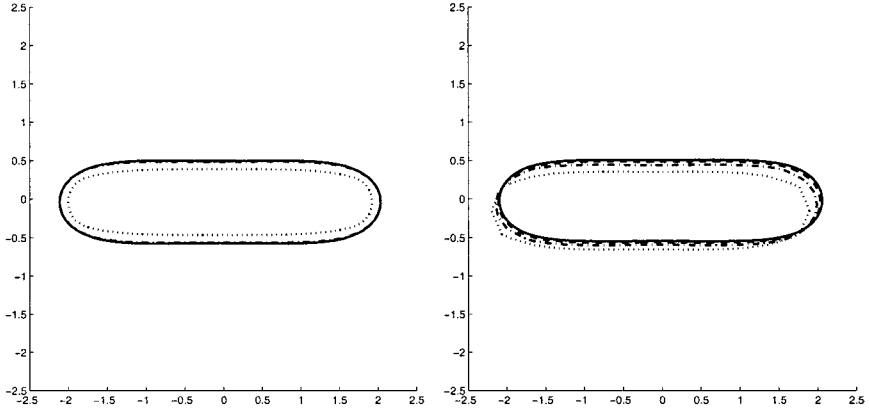

Fig. 2. Convergence check for motion by mean curvature. The figure on the left shows a convergence check for the time step. The mesh size is 0.078125 on a 128×128 grid. The dotted line is $k = 1 \times 10^{-1}$, the dash dotted line is $k = 1 \times 10^{-2}$, the dashed line is $k = 1 \times 10^{-3}$, and the solid line is $k = 1 \times 10^{-4}$. The figure on the right shows the convergence check for the mesh size. The dotted line is $h = 0.3125 (32 \times 32)$, the dash dotted line is $h = 0.15625 (64 \times 64)$, the dashed line is $h = 0.078125 (128 \times 128)$, and the solid line is $h = 0.0390625 (256 \times 256)$. The time step is 1×10^{-2}.

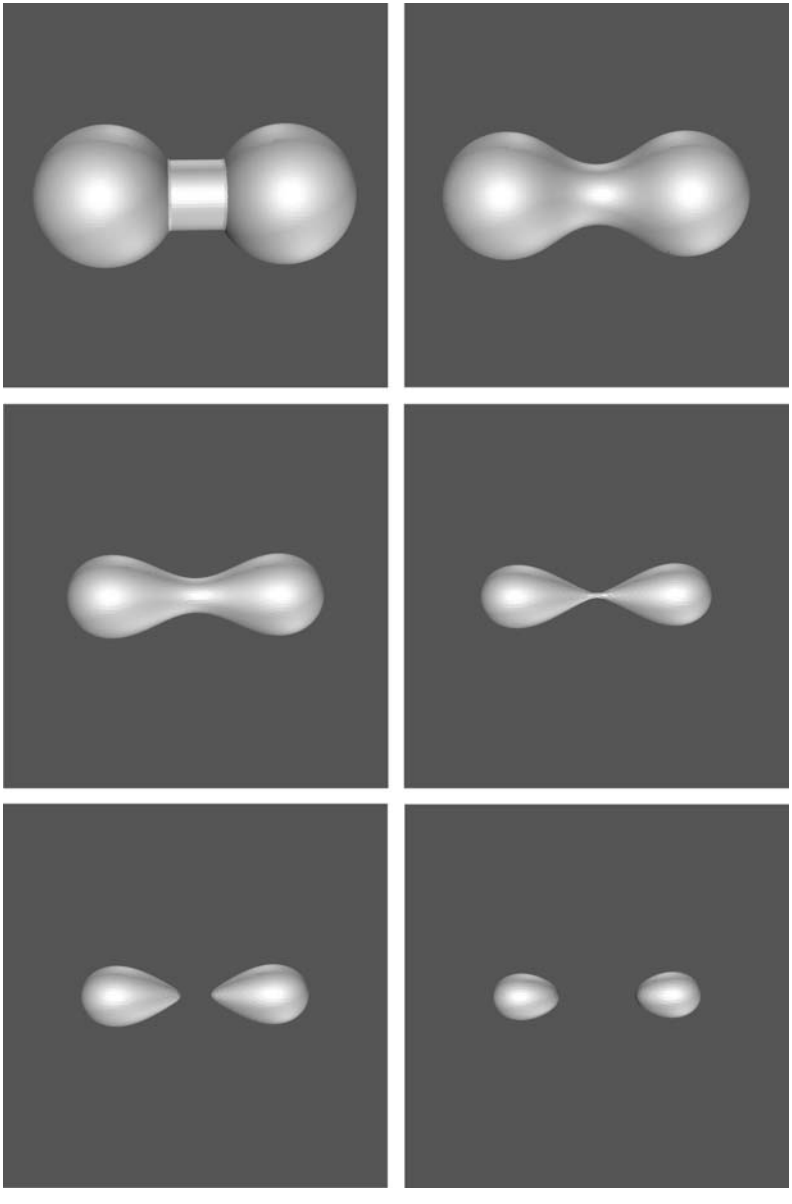

Fig. 3. Motion by mean curvature in three spatial dimensions. From the top left corner to the bottom right corner the times are $t = 0$, 0.1, 0.22, 0.70, 0.88, 0.90, and 1.0. The time step is 0.01 and the mesh size is 0.078125. The computation is done on a $128 \times 128 \times 128$ grid.

is the standard discrete form of the Laplacian based on center differencing. In Fig. 1 we present an example of this method in two spatial dimensions. Figure 2 shows a convergence check of the method. It is evident that the method converges and it stable for large time steps. Figure 3 demonstrates an example of curvature flow using this method in three spatial dimensions. In these computations we have not reinitialized level set function to be a signed distance function.

Future work will include proving that this method is stable and developing higher order accurate versions.

3. MOTION BY SURFACE DIFFUSION

In this section, guided by the previous example, we outline a semi-implicit approach for surface diffusion. We write (6) as:

$$u_t + S(u) = 0 \tag{10}$$

where $S(u) = \Delta_s \kappa \, |\nabla u|$.

Following the idea presented in Section 2 we would like to extract the dominant linear part of $S(u)$. However, given the complexity of this object we instead rewrite Eq. (10) as

$$u_t = -\beta \Delta^2 u + \beta \Delta^2 u - S(u) \tag{11}$$

where Δ^2 is the biLaplacian and β is a constant yet to be determined. Following the idea presented in Section 2 we discretize (11) in time as:

$$u^{n+1} = u^n - k\beta \Delta^2 u^{n+1} + k[\beta \Delta^2 u^n - S(u^n)]$$

where k is the time step. This equation is first order accurate in time for any value of β. The above equation can be rewritten as

$$u^{n+1} = u^n + (I + k\beta \Delta^2)^{-1} [-kS(u^n)]. \tag{12}$$

The operator $(I + k\beta \Delta^2)^{-1}$ is positive definite. In addition this operator is a smoothing operator. It is therefore apparent that we have essentially applied a smoothing operator to an explicit scheme. We have found that $\beta = 1/2$ results in a stable scheme for all time steps of interest. Therefore the smoothing operator is able to suppress the unstable high wave number modes. In addition the operator $(I + k\beta \Delta^2)^{-1}$ is easily and efficiently computed using a FFT. It appears that is idea is not new. The Laplace Modified Galerkin Method developed by Douglas and Dupont [5] uses the same idea for solving variable coefficient heat equations (see Eq. (3.2) of [5], also see [21]). This idea was also rediscovered by Glasner [6] for Hele-Shaw flows using phase field methods. Khenner *et al.* [7] suggested this idea for surface diffusion problems but did not implement it.

Remark. We note that that $\beta = 1/2$ is probably the optimal choice, which can be seen from the following simple example. Consider the o.d.e. $\dot{x} = -ax$ where $a > 0$. Applying this idea to this o.d.e. results in the difference scheme $x_{n+1} = x_n - kax_n/(1 + k\beta)$ this is stable provided $k(a - 2\beta) < 2$. It follows that the

scheme will be absolutely stable if $\beta \geqslant \frac{a}{2}$. This also shows that the term that is used to stabilize the method does not have to be the dominant linear part of $S(u)$. One could analyze $S(u)$ in more detail using the work of Xu and Zhao [21].

3.1. Surface Laplacian of the Mean Curvature

In this section we outline the computation of the surface Laplacian in two spatial dimensions. We first begin with the computation of the mean curvature, which is done as follows: we use the identity given by (7) to write the mean curvature as

$$\kappa = \frac{u_{xx}+u_{yy}}{(u_x^2+u_y^2)^{1/2}} - \frac{u_x^2 u_{xx} + 2u_x u_y u_{xy} + u_y^2 u_{yy}}{(u_x^2+u_y^2)^{3/2}}.$$

This is discretized by replacing all derivatives by center difference approximations and $u_x^2+u_y^2$ is replaced by $u_x^2+u_y^2+\varepsilon$. $\varepsilon = 10^{-8}$ was used in two spatial dimensions and $\varepsilon = 10^{-4}$ in three spatial dimensions. The choice of ε will be explained in Section 3.4.

To compute the surface Laplacian of κ, we first evaluate the surface gradient of κ as follows:

$$\nabla_s \kappa = \nabla \kappa - \mathbf{n}\partial_n \kappa.$$

In component form we have

$$\partial_n \kappa = n^x \kappa_x + n^y \kappa_y$$

where the subscripts on κ denote partial derivatives and n^x, n^y are the components of the normal vector. The derivatives κ_x and κ_y are computed using center differences and the normal vector is computed as

$$n^x = \frac{u_x}{(u_x^2+u_y^2+\varepsilon)^{1/2}} \qquad \text{and} \qquad n^y = \frac{u_y}{(u_x^2+u_y^2+\varepsilon)^{1/2}}$$

where u_x and u_y are discretized using center differences. Therefore we can write

$$\nabla_s \kappa = \kappa_x \mathbf{e}^x + \kappa_y \mathbf{e}^y - (n^x \kappa_x + n^y \kappa_y)(n^x \mathbf{e}^x + n^y \mathbf{e}^y) \equiv A\mathbf{e}^x + B\mathbf{e}^y$$

where $\mathbf{e}^x$ and $\mathbf{e}^y$ are unit vectors in the x and y direction respectively. The surface divergence is computed in an analogous fashion. We compute the surface divergence of the surface gradient to obtain the following expression for the surface Laplacian of the mean curvature,

$$\Delta_s \kappa = A_x + B_y - n^x(n^x A_x + n^y A_y) - n^y(n^x B_x + n^y B_y) \tag{13}$$

where subscripts denote partial derivatives. (13) is discretized with center differences. $S(u)$ can be computed using (13) and one has

$$S(u) = (u_x^2+u_y^2)^{1/2} \Delta_s \kappa \tag{14}$$

where u_x and u_y are approximated by center differences.

3.2. The Algorithm

Here we state the level set algorithm for motion by surface diffusion.

Step 1. Initialize u

Step 2. Set $\kappa = 0$ for all grid points.

Step 3. Compute κ at all grid points that are within one grid cell of the interface.

Step 4. Extend κ away from the interface by least 2 grid points using a velocity extension algorithm (see Appendix).

Step 5. Set $S = 0$ at all grid points.

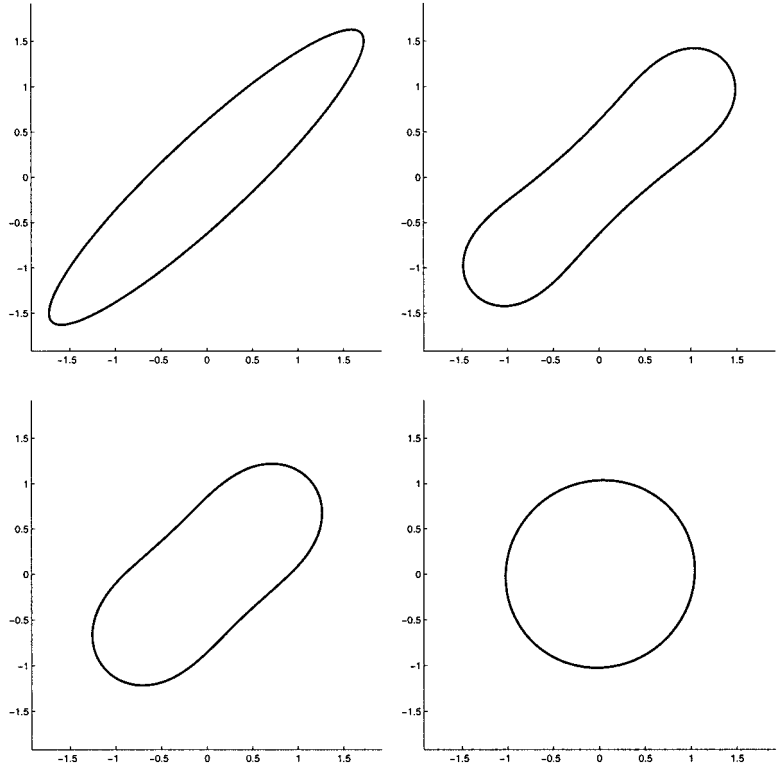

Fig. 4. Surface Diffusion in two spatial dimensions. From the top left corner to the bottom right corner the times are $t = 0$, 0.025, 0.1, and 0.4. The time step is 0.0001 and the mesh size is 0.015. The computation is done on a 256×256 grid.

Step 6. Compute S at all grid points that are within one grid cell of the interface.

Step 7. Extend S away from the interface by at least one grid point.

Step 8. Compute $(I+k\beta\Delta^2)^{-1}[kS]$ and update u using $u^{n+1}=u^n+(I+k\beta\Delta^2)^{-1}[kS]$

Step 9. One step has been completed, return to Step 2 and repeat.

Remarks. The advantage of computing κ and S near the interface and then extending them is that they can be poorly behaved just slightly away from the interface especially when a topology change has just occurred or one is imminent. In present code we have only used a first order accurate extension method. It is important to notice that we have not reinitialized u to be a distance function at any point in the computation. We also remark that Chopp and Sethian [4] also compute S near the interface and then extend it away from the interface. The major difference between this approach and their method is Step 8. Finally we remark that the operator used in Step 8 is nonlocal which can cause difficulties which will be discussed below.

3.3. Results

It has been found that $\beta=\frac{1}{2}$ has been sufficiently large to provide a stable scheme for all time steps of interest. We begin by considering the motion of an ellipse as shown in Fig. 4 where we have used a mesh size of 0.015 and a time step

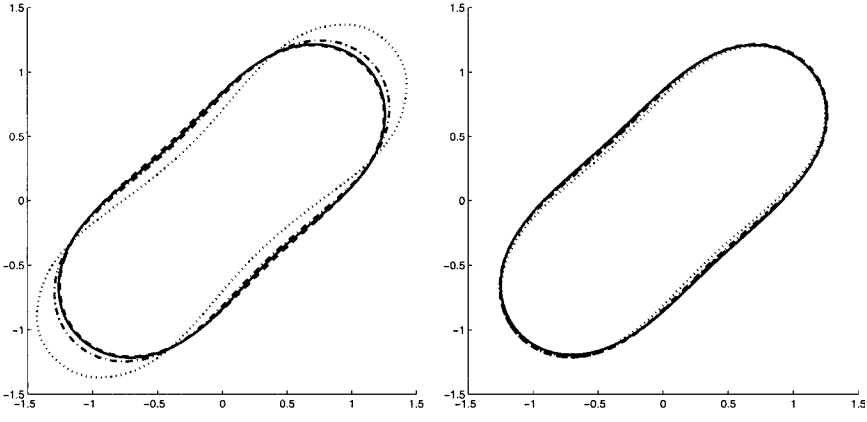

Fig. 5. Convergence check for motion by surface diffusion. The figure on the left shows a convergence check for the time step. The mesh size is 0.06 on a 64×64 grid. The dotted line is $k=5\times10^{-3}$, the dash dotted line is $k=5\times10^{-4}$, the dashed line is $k=5\times10^{-5}$, and the solid line is $k=5\times10^{-6}$. The figure on the right shows the convergence check for the mesh size. The dotted line is $h=0.12(32\times32)$, the dash dotted line is $h=0.06(64\times64)$, the dashed line is $h=0.03(128\times128)$, and the solid line is $h=0.015(256\times256)$. The time step is 1×10^{-4}.

of 1×10^{-4}. In this case $k/h^4 \approx 2 \times 10^4$ indicating a large improvement over an explicit scheme. For example in the algorithm developed by Chopp and Sethian [4] a time step of 5×10^{-6} was used with mesh size of 0.06666 ($k/h^4 \approx 0.25$). It should also be pointed out, with this same mesh size, we could have taken time steps as large as 0.005 and obtained qualitatively similar results (with less accuracy). Figure 5 presents both a spatial and temporal convergence test. It is clear that the method converges, however the rate of convergence as the time step is varied seems slow, this is consistent with the fact that this is first order accurate method. The rate of convergence as the mesh size is varied seems quite fast suggesting that the first order accurate velocity extension method did not pollute the second order accuracy of the center differences. Future work will include a careful assessment of the accuracy of this method and the development of a higher order time integration. One could use Richardson extrapolation to achieve a method that is higher order accurate in time.

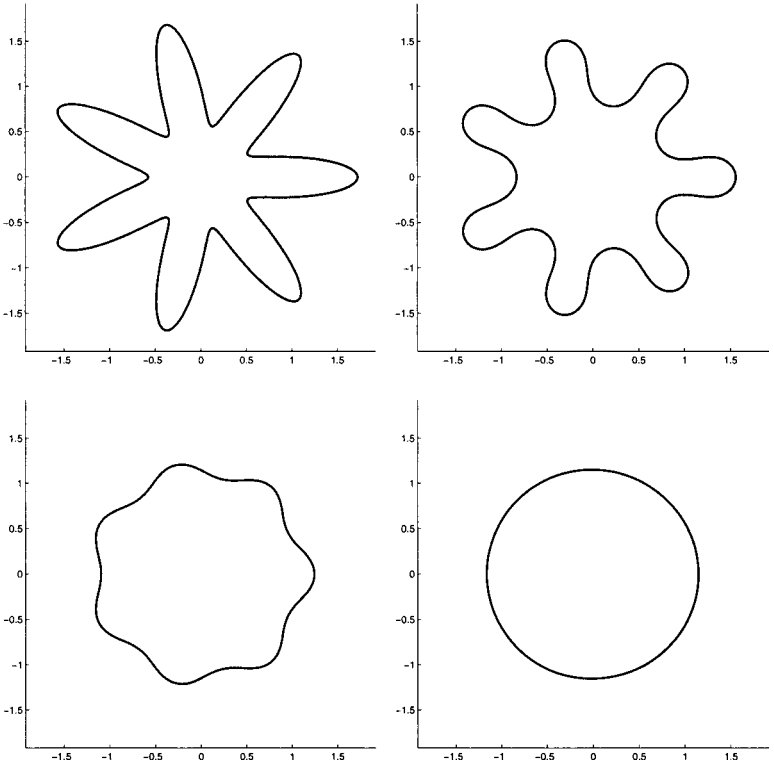

Fig. 6. Surface Diffusion in two spatial dimensions. From the top left corner to the bottom right corner the times are $t = 0$, 0.002, 0.006, and 0.08. The time step is 5×10^{-5} and the mesh size is 0.0075. The computation is done on a 512×512 grid. In the computation the area changes by less than 1%.

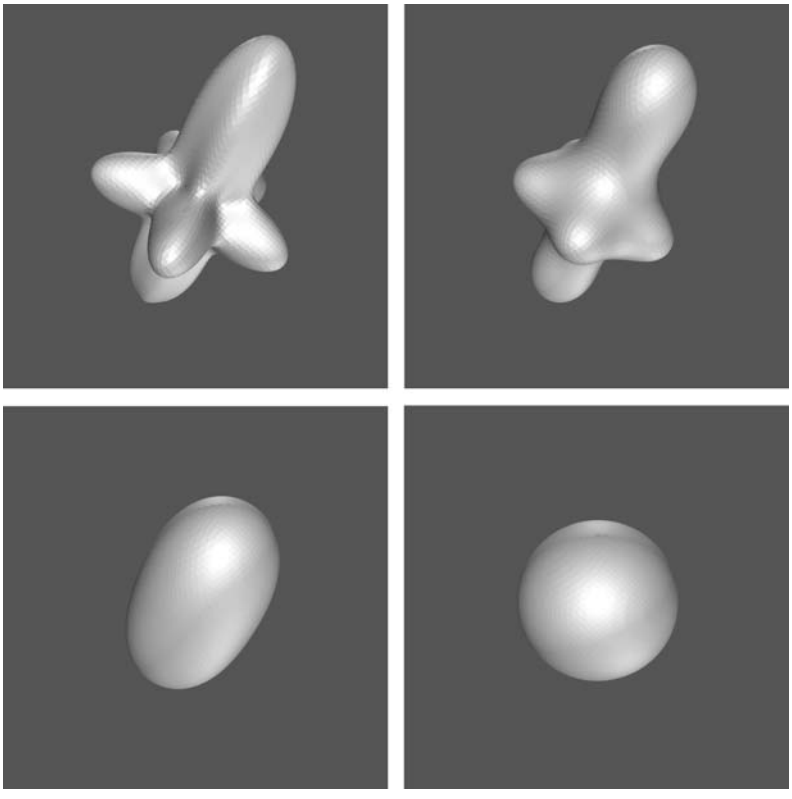

Fig. 7. Surface Diffusion in three spatial dimensions. From the top left corner to the bottom right corner the times are $t = 0$, 0.006, 0.11, and 1.0. The time step is 1×10^{-4} and the mesh size is 0.06. The computation is done on a $64 \times 64 \times 64$ grid. During the course of the computation the volume increases by approximately 3%.

Figure 6 shows the evolution of a star shaped figure into a circle. Here we have chosen a small time step for reasons of accuracy only. However this time step is still very large compared to that of an explicit method since $k/h^4 \approx 1.6 \times 10^4$. The method conserves area quite well, in this simulation the area change is less than 1%. The method converts easily to 3 dimensions and Fig. 7 shows the evolution of an irregular shape into a sphere. We observed no loss of stability in three spatial dimensions and the results in Fig. 7 would have not changed significantly if had we used a time step 10 times larger.

3.4. Topology Changes

One of the major advantages of level set methods is their ability to easily handle topological changes. However for this problem we have found this not to be the case. Given the stiffness of the problem and the lack of a maximum principle we have found merging of fronts to be somewhat difficult. This is in part because when

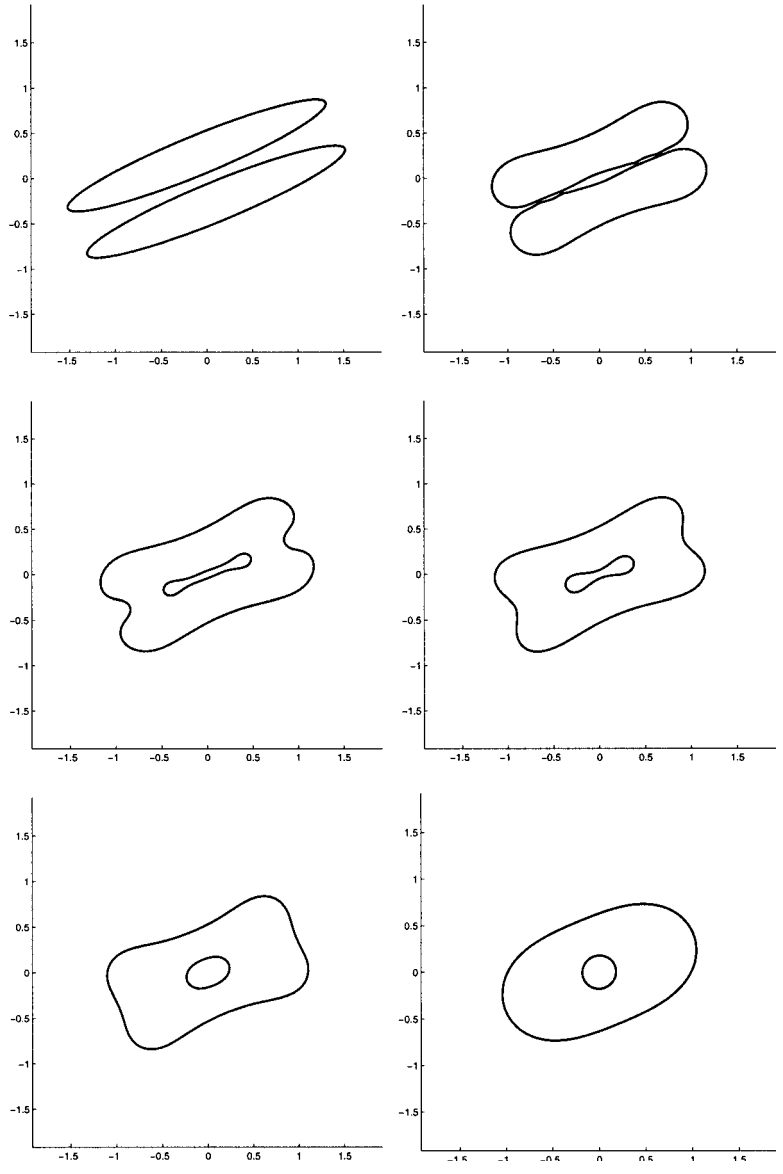

Fig. 8. Surface Diffusion in two spatial dimensions. From the top left corner to the bottom right corner the times are $t=0$, 0.0065, 0.00675, 0.007, 0.0075, and 0.025. The time step is 2.5×10^{-6} and the mesh size is 0.015. The computation is done on a 256×256 grid. In this computation the area decreased by approximately 3%. Most of the area loss occurs at the topology change.

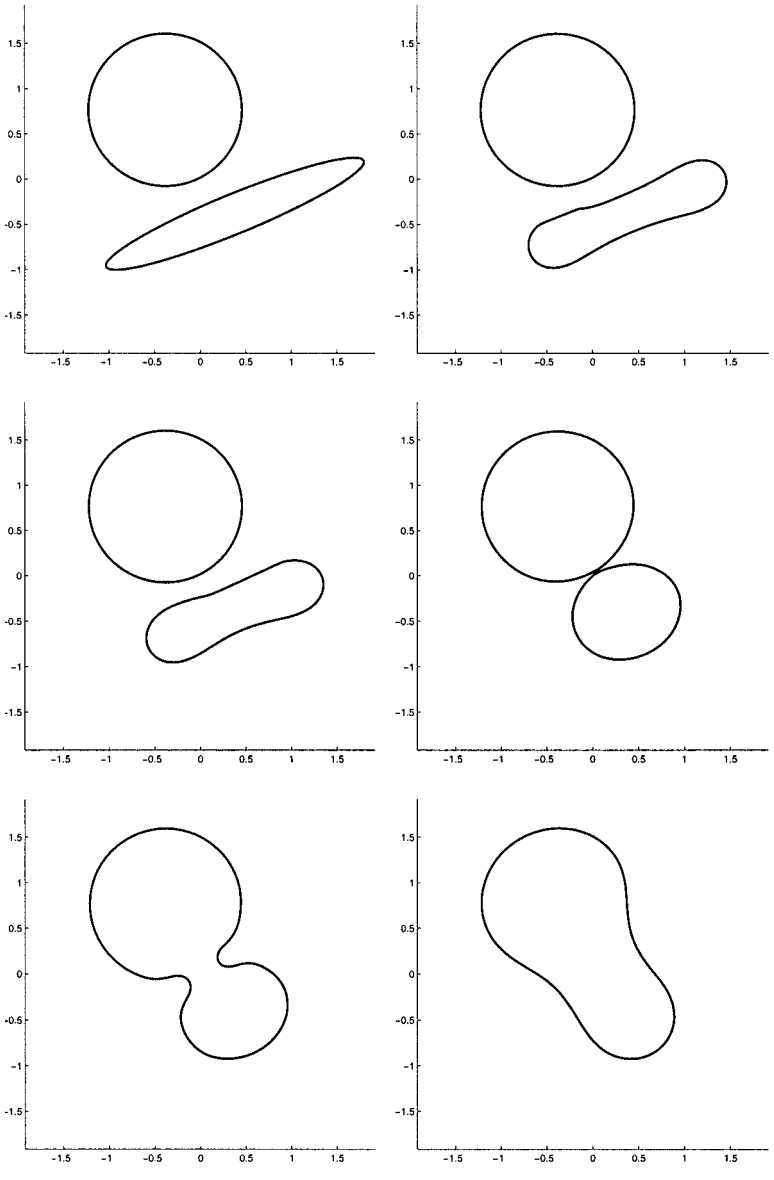

Fig. 9. Surface Diffusion in two spatial dimensions. From the top left corner to the bottom right corner the times are $t=0$, 0.005, 0.01, 0.043, 0.0435, and 0.05. The time step is 1×10^{-6} and the mesh size is 0.015. The computation is done on a 256×256 grid. In this computation the area decreased by approximately 5%.

two fronts merge the mean curvature necessarily becomes infinite and the surface Laplacian of the mean curvature is even more singular. To handle these difficulties it is important to choose ε to be sufficiently large as this tends to smooth the singularities that arise. This is why ε is chosen to be much larger for the surface diffusion problems as compared to curvature flows and why ε is larger in three spatial dimensions than in two spatial dimensions. Secondly, it is important to choose small time steps to temporally resolve the rapid transients that occur at a topology change. Figure 8 shows an example of two ellipses that merge. If we used large time steps the numerical solution would have behaved poorly at the point of merging. Figure 9 shows another example of merging. Here we start with a circle and an ellipse, the circle should not move until the ellipse grows into it. This seems to be the case; however looking closely at Fig. 9 one sees that motion of the ellipse has been slightly distorted (compare with Fig. 4). This is due to the presence of the circle and the fact that the smoothing operator is nonlocal. This interaction can be

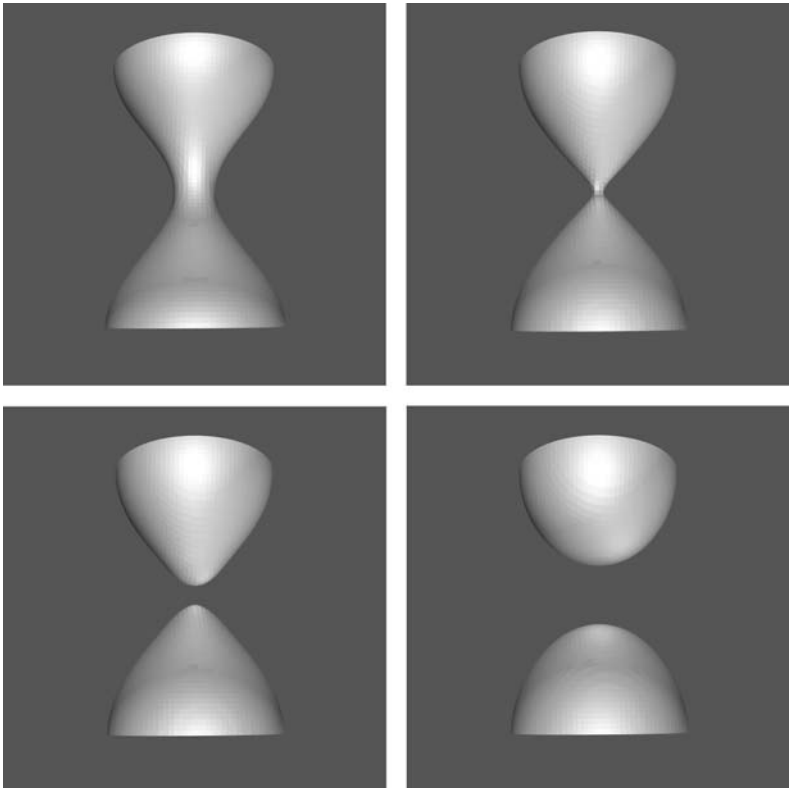

Fig. 10. Surface Diffusion in three spatial dimensions. From the top left corner to the bottom right corner the times are $t=0$, 0.031, 0.032, and 0.1. The time step is 1×10^{-4} and the mesh size is 0.06. The computation is done on a $64\times64\times64$ grid. In this computation the volume change is less than 1%.

further reduced by taking even smaller time steps. We plan to develop local smoothing operators to remove this problem in future work.

Finally, we present an example in three spatial dimensions motivated by the instability discussed by Chopp and Sethian [4]. Here we consider a sinusoidally perturbed cylinder, if the perturbation is large enough then the cylinder will pinch off in the middle. This is because the mean curvature of the neck is actually larger than the trunk. This implies that material will diffuse from the neck to the trunk thereby causing the neck to pinch off. The resulting object will evolve to sphere.

4. SUMMARY

We have introduced new level set methods for mean curvature flow and motion by surface diffusion. These methods are based on operator splitting and are semi-implicit. Their main advantage is that they are as simple to implement as an explicit method but it is possible to take much bigger time steps (by a factor of 10^2 to 10^4). In the case of surface diffusion our method amounts to applying a smoothing operator to an explicit method. The smoothing operator is nonlocal but can be easily computed using FFTs. The nonlocal nature of this operator can produce some difficulties; in future work we plan to develop local smoothing operators.

APPENDIX–EXTENSION ALGORITHM

In this section we outline the extension algorithm. The basic idea is to solve a hyperbolic equation whose characteristics point outward from the interface and thereby carry information near the interface to the rest of the domain. This idea is very similar to the construction of a distance function developed by Sussman *et al.* [19]. The idea of extending quantities off the front in this fashion was developed by Chen, Merriman, Osher, and Smereka in 1993 and was subsequently used in their work on a level set method for the Stefan problem [3]. The method also appears in Susan Chen's Ph.D. dissertation [2]. In their work a one dimensional version was used (see [2, 3] for more details). The form used in this paper was introduced by Zhao *et al.* [22].

In this setting u is the level set function and c is a variable that is known near the zero level set of u. We wish to extend c off the interface in such away it is constant in normal directions. This is done by solving the following hyperbolic equation:

$$\frac{\partial c}{\partial t} + \text{sign}(u)\frac{\nabla u}{|\nabla u|}\cdot\nabla c = 0. \tag{15}$$

We outline the numerical discretization of (15) in one space dimension. Extensions to two and three dimensions are straightforward. To begin we define the backward and forward, first order accurate derivatives of the level set function

$$D_- u_j^n = \frac{u_i^n - u_{i-1}^n}{h}, \qquad D_+ u_j^n = \frac{u_{i+1}^n - u_i^n}{h}$$

and the Godunov-type upwind derivative

$$D^g u_j^n = m([D_- u_j^n]^+, [D_+ u_j^n]^-) \qquad \text{where} \quad m(x, y) = \begin{cases} x & |x| > |y| \\ y & \text{otherwise,} \end{cases}$$

$[q]^+$ denotes the positive part of q and $[q]^-$ denotes the negative part. Next we define the upwind derivatives for c as

$$Dc_j^n = \begin{cases} D_- c_j^n & \text{if} \quad D^g u_j^n > 0 \\ D_+ c_j^n & \text{otherwise.} \end{cases}$$

Finally we define the discrete sign function as

$$s(u_j^n) = \begin{cases} \dfrac{u_j^n}{h(|D^g u_j^n| + \varepsilon)} & \text{if} \quad u_j^n u_{j+1}^n \leqslant 0 \qquad \text{or} \qquad u_j^n u_{j-1}^n \leqslant 0 \\ \text{sign}(u_j^n) & \text{otherwise.} \end{cases} \tag{16}$$

The discrete form of (15) is:

$$c_j^{n+1} = c_j^n - ks(u_j^n)\, D^g u_j^n Dc_j^n / (|D^g u_j^n| + \varepsilon).$$

Equation (16) is very close to the sign function introduced by Russo and Smereka [14].

ACKNOWLEDGMENTS

I thank R. Nochetto for pointing out [5] to me. This work was supported by the NSF through a Career Award and Grant number DMS-0207402.

REFERENCES

1. Adalsteinsson, D., and Sethian, J. A. (1997). A unified level set approach to etching, deposition, and lithography III: Complex simulations and multiple effects. *J. Comput. Phys.* **138**(1), 193–223.
2. Chen, S. (June 1996). Ph.D. Dissertation, Dept. of Mathematics, UCLA.
3. Chen, S., Merriman, B., Osher, S., Smereka, P. (1997). A simple level set method for solving Stefan problems. *J. Comput. Phys.* **135**, 8–29.
4. Chopp, D. L., and Sethian, J. A. (1999). Motion by intrinsic Laplacian of curvature. *Interfaces and Free Boundaries* **1**, 107.
5. Douglas, J., Jr., and Dupont, T. (1971). Alternating-direction galerkin methods on rectangles. In Hubbard, B. (ed.), *Numerical Solution of Partial Differential Equations*, Academic Press, New York.
6. Glasner, K. A diffuse interface approach to Hele-Shaw flow. Preprint 2002.
7. Khenner, M., Averbuch, A., Israeli, M., and Nathan, M. (2001). Numerical simulation of grain-boundary grooving by level set method. *J. Comput. Phys.* **170**, 764–784.
8. Li, Z. L., Zhao, H., and Gao, H. J. (1999). A numerical study of electro-migration voiding by evolving level set functions on a fixed Cartesian grid. *J. Comput. Phys.*, **152**, 281.
9. Mullins, W. W. (1957). Theory of thermal grooving. *J. Appl. Phys.* **28**, 333.
10. Mullins, W. W. (1995). Mass transport at interfaces in single component systems. *Metall. Mater. Trans.* **26A**, 1917

11. Merriman, B., Bence, J., and Osher, S. (1994). Motion of multiple junctions: A level set approach. *J. Comput. Phys.* **112**, 334.
12. Osher, S. J., and Fedkiw, R. P. (2001). Level set methods: An overview and some recent results. *J. Comput. Phys.* **169**, 463–502.
13. Osher, S., and Sethian, J. A. (1988). Fronts propagating with curvature dependent speed: Algorithms based on Hamilton–Jacobi Formulations. *J. Comput. Phys.* **79**, 12.
14. Russo, G., and Smereka, P. (2000). A remark on computing distance functions. *J. Comp. Phys.* **163**, 51–67.
15. Ruuth, S. J. (1998). Efficient algorithms for diffusion-generated motion by mean curvature. *J. Comput. Phys.* **144**, 603–625.
16. Sethian, J. A. (1996). *Level Set Methods: Evolving Interfaces in Geometry, Fluid Mechanics, Computer Vision, and Materials Sciences*, First Edn., Cambridge University Press.
17. Sethian, J. A., and Smereka, P. (2003). Level set methods for fluid interfaces. *Ann. Rev. Fluid Mech.*
18. Srolovitz, D. J., Mazor, A., and Bukiet, B. G. (1988). Analytical and numerical modeling of columnar evolution in thin films. *J. Va. Soc. Technol. A* **6**, 2371.
19. Sussman, M., Smereka, P., and Osher, S. (1994). A level set approach for incompressible two-phase flow. *J. Comp. Phys.* **114**, 146–159.
20. Tasdizen, T., Whitaker, R., Burchard, P., and Osher, S. *Geometric Surface Processing via Normal Maps*, UCLA CAM Report 02-03.
21. Xu, J., and Zhao, H. *An Eulerian Formulation for Solving Partial Differential Equations along a Moving Interface*, UCLA CAM report 02-27.
22. Zhao, H. K., Chan, T., Merriman, B., Osher, S. J. (1996). A variational level set approach to multiphase motion. *J. Comput. Phys.* **127**, 179–95.