JOURNAL OF COMPUTATIONAL PHYSICS **134**, 236–252 (1997)
ARTICLE NO. CP975689

A Hybrid Method for Moving Interface Problems with Application to the Hele–Shaw Flow[1]

Thomas Y. Hou,*[,2] Zhilin Li,†[,3] Stanley Osher,†[,4] and Hongkai Zhao†[,5]

Applied Mathematics, California Institute of Technology, Pasadena, California 91125; †Department of Mathematics, University of California at Los Angeles, Los Angeles, CA 90095.

Received April 11, 1996; revised January 10, 1997

In this paper, a hybrid approach which combines the *immersed interface method* with the *level set approach* is presented. The fast version of the immersed interface method is used to solve the differential equations whose solutions and their derivatives may be discontinuous across the interfaces due to the discontinuity of the coefficients or/and singular sources along the interfaces. The moving interfaces then are updated using the newly developed fast level set formulation which involves computation only inside some small tubes containing the interfaces. This method combines the advantage of the two approaches and gives a second-order Eulerian discretization for interface problems. Several key steps in the implementation are addressed in detail. This new approach is then applied to Hele–Shaw flow, an unstable flow involving two fluids with very different viscosity. © 1997 Academic Press

1. INTRODUCTION

Many physically interesting problems involve propagation of moving interfaces. Vortex sheet rollup in hydrodynamic instability, wave interactions on the ocean's free surface, solidification in crystal growth, and Hele–Shaw cells for pattern formation are some of the better known examples. Typically, these interface problems are very singular and are sensitive to small perturbations. Consequently, it is very challenging to obtain accurate and stable numerical approximations for these moving interface problems.

There are two basic numerical approaches to interface problems. One is based on front tracking, the other on front capturing. Both approaches have their advantages and disadvantages. In front tracking methods, one can design accurate approximations to the moving front without differentiating across the front. Boundary integral methods are examples of this type; see, e.g., the recent review paper [12] for references. The disadvantage of front tracking methods is that it requires explicit tracking of the front. This is, in general, difficult for interfaces with complicated geometry and topological change and particular so in three dimensions. Front capturing, in particular, the level set method as derived by Osher and Sethian in [26], on the other hand, avoids the explicit tracking of the front. The moving front is implicitly captured on an Eulerian grid. As a consequence, complex interface structures and topological changes can be captured quite naturally in two and three dimensions; see, e.g., [4, 26, 32, 33]. One difficulty associated with the conventional front capturing approach is the possible loss of high order accuracy at the moving front. This is especially the case for incompressible fluid interfaces with surface tension [3, 4, 33], but also exists in numerical methods using front tracking; see, e.g. [28, 36], where the effect of surface tension is modeled by a singular delta function source term. The immersed interface method developed by LeVeque and Li [15] attempts to use a semi-Eulerian method to achieve a uniformly high order accuracy up to the free surface, by incorporating the jump condition at the moving interface into the discretization. However, their method requires explicit information about the moving interface. This makes it difficult to apply to general free boundary problems with complicated geometry or topology.

This paper attempts to combine the advantages of the immersed interface method and the level set approach. This gives rise to a completely Eulerian front capturing method with uniformly high order accuracy up to the moving interface. The main idea is as follows. For fluid interface problems, surface tension introduces a jump condition in the pressure and/or its normal derivative across the moving interface. If we incorporate these jump conditions in our finite difference discretization across the moving interface, we can derive a uniformly high order discretization up to the moving interface. In the original version of the immersed interface method, the moving interface is either given *a priori* or is tracked explicitly. In the hybrid im-

[1] This work was supported by URI ARPA-ONE Grant N00014092-J-1890, NSF Grants DMS-94-04942, DMS-94-07030, and DMS-96-26703, and DOE Grant DE-FG03-89ER25073.

[2] E-mail: hou@ama.caltech.edu.
[3] E-mail: zhilin@math.ucla.edu.
[4] E-mail: sjo@math.ucla.edu.
[5] E-mail: hzhao@math.ucla.edu.

mersed interface/level set method proposed in this paper, we will use an Eulerian level set function to capture the interface. The information regarding the location and the local normal vector can be extracted from this level set function. When we incorporate this information into the immersed interface discretization, we obtain a uniformly high order discretization. Clearly we get the advantages of both methods and avoid the shortcomings of these two methods. This gives rise to a robust and accurate Eulerian discretization for interface problems.

In order to test the robustness of the method, we apply our method to compute the expanding Hele–Shaw bubble problem. In this problem, a less viscous and immiscible fluid is injected into a more viscous flow. This is an important test problem because it can be used as a model to study pattern formation in crystal growth and solidification. This is a very challenging numerical problem due to the underlying Mullins–Sekerka instability. Small perturbations, even at the level of round-off errors, can lead to rapid growth for those unstable modes in high wave numbers, especially for the case of small surface tension. For this reason, it is essential to have a stable numerical discretization. In the case of boundary integral calculations, a Fourier filtering technique has been used to control the artificial growth of the round-off errors [7, 13, 14]. In the case of level set methods, we use a re-initialization process to remove the high frequency instability. This re-initialization process was first introduced by Sussman, Smereka, and Osher in [33] to maintain the level set function as a signed distance function. It plays the role of geometrical regularization. In effect, it produces a cutoff to high frequency noise. We have performed a careful numerical study and have not observed any numerical instability of our method as we refine the mesh. On the other hand, the re-initialization process as well as the basic finite difference approximation to the level set motion must introduce some numerical dissipation. This also places a limitation to the resolution of the smallest scales in the physical problem. In our numerical study, we have performed a careful resolution study to quantify the relationship between the effect of numerical dissipation and the effect of surface tension.

Our numerical experiments also suggest that while high frequency noise has been damped by numerical dissipation, numerical noise at low frequency components still persists. This numerical noise is more difficult to filter than the high frequency noise since it is mixed with the low frequency components. For small surface tension, there are many unstable modes. In this case, the low frequency noise due to roundoff errors will produce un-symmetric patterns in long time numerical simulations, even if we start with a symmetric initial configuration. This seems to be consistent with experimental observations. As is common to most grid-based methods, we also observe some grid effect for long time calculations. This is due to the nonlinear nature of our finite difference discretization and the way in which the discretization is modified near the interface. However, we show that when the physical solution is well resolved, the grid effect is almost negligible.

There are two advantages of our hybrid method. The first one is that it gives an Eulerian discretization that is uniformly second-order accurate up to the moving interface. The second advantage is that the method is very fast. The improvement in speed is due to two factors. The first one is due to a fast version of the level set method [39]. In this version of the level set method, only a small region containing the moving interface need to be updated in time. This basically reduces the number of operations to $O(N)$, where N is the number of grid points along the moving interface. The second factor is more crucial. A preconditioned fast immersed interface method [20] is used in discretizing the pressure equation. This effectively reduces the discrete pressure equation to a discrete Poisson equation. Thus a standard fast Poisson solver can be used to speed up the calculation. The operation count is only of order $O(M^2)$, where M is the number of Eulerian grid points in each dimension. This gives a much faster method than solving the original pressure equation with discontinuous coefficients and large jumps.

The rest of the paper is organized as follows. In Section 2, we describe the immersed interface method and its fast version developed recently. In Section 3, we describe the level set method and its fast version. The question of how to reconstruct the interface from a level set function will be discussed in detail. Section 4 presents our hybrid method to the Hele–Shaw problem. Some numerical issues, such as re-initialization of the level set function and smooth extension of the interface velocity outside the interface will be discussed. We present detailed numerical experiments in Section 5. We give a concluding remark in Section 6.

2. THE IMMERSED INTERFACE METHOD

In this section, we will review the main idea of the immersed interface method originally developed by LeVeque and Li in [15, 19]. The *immersed interface method* provides an effective discretization for differential equations with discontinuous coefficients or singular source terms across a free interface. Due to the discontinuity or singular source term in the equation, the solution and/or its derivatives may become discontinuous across the interface. The main idea is to incorporate the known jump conditions in the solution and its derivatives, e.g., the flux across the interface, into the finite difference scheme. This gives rise to a modified scheme on a Cartesian grid. Numerical tests have shown that solutions obtained from this method are second-order accurate at all points as long as the interface remains smooth. This approach has also been applied to three-dimensional elliptic equations [22], parabolic equations [21, 23], hyperbolic wave equations with discontinuous coefficients [17], and incompressible Stokes flow problems with moving interfaces [16, 18].

As a brief example to show how the immersed interface method works, we consider a Poisson equation

$$\Delta u = \int_{\Gamma} C(s)\delta(x - X(s))\delta(y - Y(s))\, ds, \qquad (x, y) \in \Omega, \tag{2.1}$$

with given boundary conditions on the boundary $\partial\Omega$, where $(X(s), Y(s))$ is the arc length parameterization of the interface Γ. With the immersed interface method, the problem can be written as

$$\Delta u = 0, \qquad (x, y) \in \Omega - \Gamma, \tag{2.2}$$

$$[u] = 0, \qquad [u_n] = C(s), \tag{2.3}$$

where $[u]$ and $[u_n]$ are the jumps of the solution and its normal derivative across the interface Γ, respectively. The discrete form of (2.1) is simply the standard five-point scheme plus a correction term at irregular grid points where the interface cuts through the five-point stencil,

$$\frac{U_{i+1,j} + U_{i-1,j} + U_{i,j+1} + U_{i,j-1} - 4U_{ij}}{h^2} = f_{ij} + C_{ij}. \tag{2.4}$$

Thus a fast Poisson solver such as a fast Fourier transformation method (FFT), cyclic reduction, etc. (see [34]) can be employed if the solution domain is a rectangle.

Solving an elliptic equation in the form

$$\nabla(\beta(x, y)\nabla u) = f(x, y), \tag{2.5}$$

where the coefficient β is discontinuous across the interface(s), is a major part in many computational fluid dynamics problems (see [4, 11, 33] etc.), It is also one of the two governing equations in the simulation of Hele–Shaw flow (see Section 4). For the Hele–Shaw flow, the solutions itself is discontinuous across the interface which corresponds to a dipole source along the interface. Equation (2.5) is only valid in the interior of the domain, but not on the interface. To solve Eq. (2.5), we cannot use a fast Poisson solver with the original immersed interface even if β is a piecewise constant. Thus better iterative methods are sought to solve the linear system of equations resulting from the discretization of *the immersed interface method.* One such method is the multigrid method developed by Adams [2]. The approach used here is based on a *preconditioning approach* [20]. We will briefly discuss it in the following.

2.1. *The Fast Immersed Interface Method for Elliptic Equations with Piecewise Constant Coefficients*

The main idea of the fast immersed interface method is to precondition the differential equation before applying the immersed interface method. An intermediate unknown function, which describes the jump in the normal derivative across the interface, is introduced. The discretization is equivalent to using a second order difference scheme for the regular grid points in the region, and a second order discretization for a Neumann like interface boundary condition. The resulting discretization satisfies the maximum principle, and the solution is second order accurate globally based on conventional analysis [25]. Below we explain the main idea by considering the following elliptic interface problem.

Problem I.

$$\nabla(\beta(x, y)\nabla u) = f(x, y), \tag{2.6a}$$

$$\text{given } BC \text{ on } \partial\Omega, \tag{2.6b}$$

with specified jump conditions along the interface $\Gamma(s)$

$$[u] = w(s), \tag{2.7a}$$

$$[\beta u_n] = v(s), \tag{2.7b}$$

where s is the arc length of the interface.

With the original immersed interface method, we are able to derive a difference scheme for which the local truncation error is $O(h^2)$ away from the interface, and $O(h)$ near the interface, with a six-point stencil. However, if the jump in the coefficient β is very large as in the case of the Hele–Shaw flow, the difference scheme may lose the sign property required for the maximum principle because the scheme has to be modified to satisfy the flux condition (2.7b). On the other hand, if we can use the jump condition in the normal derivative $[u_n]$, which is an unknown in Problem I, then the modified difference scheme will satisfy the maximum principle. Thus we consider the solution $u_g(x, y)$ of the following problem, which depends on a newly introduced function $g(s)$ describing the jump in the normal derivative.

Problem II.

$$\Delta u + \frac{\nabla\beta^+}{\beta^+} \cdot \nabla u = \frac{f}{\beta^+}, \quad \text{if } x \in \Omega^+,$$

$$\Delta u + \frac{\nabla\beta^-}{\beta^-} \cdot \nabla u = \frac{f}{\beta^-}, \quad \text{if } x \in \Omega^-, \tag{2.8a}$$

$$\text{given } BC \text{ on } \partial\Omega, \tag{2.8b}$$

with specified jump conditions

$$[u] = w(s), \tag{2.9a}$$

$$[u_n] = g(s). \tag{2.9b}$$

Notice that the jump conditions (2.9a) and (2.9b) depend on the singularities of the source term $f(x, y)$ along the interface. For example, if $f(x, y)$ contains the Dirac delta function distribution along the interface, then the source strength will contribute to the jump in the normal derivative across the interface. However, in the expression of (2.8a), we do not need information about $f(x, y)$ on the interface Γ, so there is no need to write $f(x, y)$ differently. Let the solution of Problem I be $u^*(x, y)$, and define

$$g^*(s) = [u_n^*](s) \tag{2.10}$$

along the interface Γ. Then $u^*(x, y)$ satisfies the elliptic equation (2.8a)–(2.8b) and the jump conditions (2.9a)–(2.9b) with $g(s) \equiv g^*(s)$. In other words, $u_{g^*}(x, y) \equiv u^*(x, y)$, and

$$\left[\beta \frac{\partial u_{g^*}}{\partial n}\right] = v(s) \tag{2.11}$$

is satisfied. Therefore, solving Problem I is equivalent to finding the corresponding $g^*(s)$ and then $u_{g^*}(x, y)$ in Problem II. Notice that $g^*(s)$ is only defined along the interface. Thus it is defined in a space one dimension lower than that of $u(x, y)$.

For simplicity, consider the special case where β is a piecewise constant. Now we have $\nabla\beta^- = 0$ and $\nabla\beta^+ = 0$. It is very easy to discretize Problem II using the immersed interface method. We have the standard discrete Laplacian operator for Δu plus some correction terms. The correction terms are the functions of the jumps $[u]$, $[u_x]$, $[u_y]$, $[u_{xx}]$, $[u_{yy}]$, which are all known if we know $[u]$ and $[u_n]$ along the interface. The jump condition (2.7b) then is discretized using our new technique called *the weighted least squares interpolation* [20], which is very robust and second-order accurate for interface problems. Thus the discrete form of our approach can be written as the following linear system,

$$\begin{bmatrix} A & B \\ E & D \end{bmatrix} \begin{bmatrix} U \\ G \end{bmatrix} = \begin{bmatrix} F \\ V \end{bmatrix}. \tag{2.12}$$

The solution U and G are the discrete form of the solution $u_{g^*}(x, y)$ and $g^*(s)$, the solution of Problem II which satisfies (2.11). Eliminating U from (2.12) gives a linear system for G

$$\begin{aligned}(D - EA^{-1}B)G &= V - EA^{-1}F \\ &\overset{\text{def}}{=} \overline{V}.\end{aligned} \tag{2.13}$$

This is a much smaller linear system compared to the one for U. The coefficient matrix is the Schur complement of D in (2.12). In practice, the matrices A, B, ..., and the vectors $\overline{V}$, F are never explicitly formed. The matrix and vector are written above merely for theoretical purposes. Thus an iterative method, such as GMRES iteration [30], is preferred. With this approach, the number of iterations for solving the Schur complement system of (2.13) is independent of the jump $[\beta]$ and the mesh size h. The details of the algorithms can be found in our recent work [20] for piecewise constant β, where we can take advantage of fast Poisson solvers. For variable coefficients, we plan to use the multigrid method developed by Adams [2].

3. THE LEVEL SET METHOD

Once we obtain a second-order discretization of the interface problem in space, we need to update the interface in time. Three possible numerical algorithms are the "volume of fluid" technique, the marker particle approach, and the level set formulation. The level set formulation originating in [26] is the one we will use in this paper.

We will concentrate on the two-phase flow problems here, although similar techniques can be, and actually have been, applied to multiphase problems [38]. Suppose there is a closed interface separates the less viscous flow from more viscous one. Let $\Gamma(t)$ be the moving interface, $\Omega^-(t)$ and $\Omega^+(t)$ be the interior and exterior regions of the interface respectively. The moving interface $\Gamma(t)$ can be described as the zero level set of a function $\varphi(x, y, t)$, which is Lipschitz continuous, satisfying

$$\varphi(x, y, t) > 0 \quad \text{for } (x, y) \in \Omega^-, \tag{3.14a}$$

$$\varphi(x, y, t) = 0 \quad \text{for } (x, y) \in \Gamma, \tag{3.14b}$$

$$\varphi(x, y, t) < 0 \quad \text{for } (x, y) \in \Omega^+. \tag{3.14c}$$

Therefore, by differentiating the level set $\varphi(x, y, t) = c$ with respect to time t, we can obtain the equation of motion of the level set

$$\varphi_t + \mathbf{u} \cdot \nabla\varphi = 0. \tag{3.15}$$

This is referred to as the level set equation of Hamilton–Jacobi type. Initially, $\varphi(x, y, 0)$ is often chosen as the signed normal distance from the interface which means $|\nabla\varphi| = 1$. By solving the modified Hamilton–Jacobi equation, we can update the moving interface, the zero level set of $\varphi(x, y, t) = 0$.

3.1. *Reconstruction of the Interface from the Level Set Function*

In the level set representation, the interface, which is the set of points (x, y) satisfying $\varphi(x, y) = 0$, is not explicitly

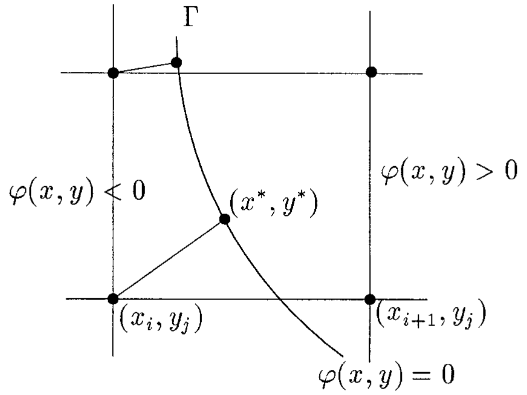

FIG. 1. Finding the control point $\mathbf{X}^*$ from an irregular grid point (x_i, y_j), $\varphi(x_i, y_j) \le 0$.

given.[1] Instead we only have information $\varphi(x_i, y_j)$ at each grid point. In order to use the fast immersed interface method [20], we need to find a number of control points on the interface so that we can set up equations for the intermediate unknowns $[u_n]$. There are two criteria in choosing those control points:

- We want the process to be *local*. One of the advantages of the level set approach is that the geometric characteristics of the interface are completely determined by the level set function $\varphi(x_i, y_j)$. We need to preserve such local properties in determining the control points.
- We do not want to have *clustered* control points to avoid unnecessary large and ill-conditioned system (2.13).

We now describe a practical *reconstruction* process which meets the requirements above.

Consider an irregular grid point $\mathbf{X} = (x_i, y_j)$, where the interface cuts through the standard five point stencil centered at (x_i, y_j), on a particular side, say $\varphi(x, y) \le 0$. Using the following process, we can find the *projection* on the interface (Fig. 1):

1. Find the unit steepest ascent direction $\mathbf{p}$ at $\mathbf{X}$

$$\mathbf{p} = \frac{\nabla \cdot \varphi}{\|\nabla \cdot \varphi\|}. \tag{3.16}$$

2. Locate the projection of $\mathbf{X}$ on the interface along the direction p:

$$\mathbf{X}^* = \mathbf{X} + \alpha \mathbf{p}, \tag{3.17}$$

where α is determined from the following quadratic equation:

$$\varphi(\mathbf{X}) + \|\nabla \cdot \varphi\|\,\alpha + \tfrac{1}{2}\left(\mathbf{p}^T He(\varphi)\mathbf{p}\right)\alpha^2 = 0, \tag{3.18}$$

where $He(\varphi)$ is the Hessian matrix of φ,

$$He(\varphi) = \begin{bmatrix} \varphi_{xx} & \varphi_{xy} \\ \varphi_{yx} & \varphi_{yy} \end{bmatrix}, \tag{3.19}$$

evaluated at $\mathbf{X}$.

If we repeat this process for each irregular grid point on the side of $\varphi(x, y) \le 0$, we can get the set of control points representing the interface with second-order accuracy. To use the fast immersed interface method we still need to compute the normals, tangents, and curvatures on the interface, or at the control points. This can be done using the level set information again. For example, the unit normal, the unit tangent, and the curvature at a point can be expressed as

$$\mathbf{n} = -\frac{\nabla \cdot \varphi}{\|\nabla \cdot \varphi\|} = -\left[\frac{\varphi_x}{\sqrt{\varphi_x^2 + \varphi_y^2}}, \frac{\varphi_y}{\sqrt{\varphi_x^2 + \varphi_y^2}}\right]^T, \tag{3.20}$$

$$\mathbf{t} = \left[-\frac{\varphi_y}{\sqrt{\varphi_x^2 + \varphi_y^2}}, \frac{\varphi_x}{\sqrt{\varphi_x^2 + \varphi_y^2}}\right]^T, \tag{3.21}$$

$$\kappa = -\frac{\varphi_{xx}\varphi_y^2 - 2\varphi_{xy}\varphi_x\varphi_y + \varphi_{yy}\varphi_x^2}{(\varphi_x^2 + \varphi_y^2)^{3/2}} = -\nabla \cdot \left(\frac{\nabla \varphi}{\|\nabla \cdot \varphi\|}\right). \tag{3.22}$$

To get these quantities, we need to find accurate values of φ, φ_x, φ_y, φ_{xx}, φ_{xy}, and φ_{yy} at some point on the interface which may not lie on a grid point. This can be done through the bilinear interpolation described below. Note that at a grid point, the information can be calculated through central difference approximations unless there is a singularity of φ in the neighborhood of that grid point (see Fig. 2).

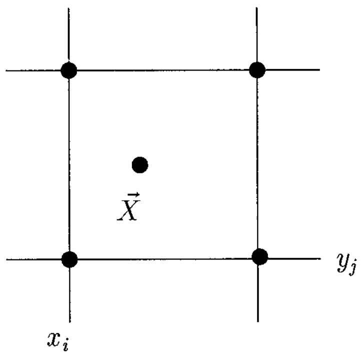

FIG. 2. A diagram for the bilinear interpolation.

[1] We will omit the time dependence of the level set function $\varphi(x, y, t)$ for simplicity if no confusion occurs.

Given any point (x, y), we can find the square which contains the point (x, y) but no other grid points except the four vertices, say, (x_i, y_j), (x_{i+1}, y_j), (x_i, y_{j+1}), and (x_{i+1}, y_{j+1}). Let G_{ij} be a grid function which approximates one of the quantities, such as φ_x, φ_y, etc., in (3.20)–(3.22). We can use the following bilinear interpolation to get an approximation of $G(x, y)$,

$$G(x,y) = \frac{1}{4} \sum_{k=0,l=0}^{1} G_{i+k,j+l}\, \bar{x}_k\, \bar{y}_l, \tag{3.23}$$

where

$$\bar{x}_k = 1 + (2k-1)\left(\frac{2(x-x_i)}{h} - 1\right),$$
$$\bar{y}_l = 1 + (2l-1)\left(\frac{2(y-y_j)}{h} - 1\right).$$

Using the bilinear interpolation formula described above, we can evaluate all the quantities in (3.20)–(3.22) at any point in terms of the corresponding values at neighboring grid points.

3.2. *Re-initialization*

Initially, $\varphi(x, y, 0)$ is often chosen as the signed normal distance from the interface which means $|\nabla\varphi| = 1$. By solving the modified Hamilton–Jacobi equation, we can update the moving interface, the zero level set of $\varphi(x, y, t) = 0$. However, while Eq. (3.15) will move the interface at the correct speed, φ in general is no longer a distance function. In fact, φ develops steep or flat gradients, especially when topological changes such as breaking and merging take place, or when the velocity field near the interface is singular. This difficulty can be avoided by a re-initialization process introduced in [32, 33] so that φ will remain as the signed distance function up to a certain accuracy. The new level set function φ is the steady state solution of the equations

$$\varphi_t + (|\nabla\varphi| - 1)H(\varphi) = 0, \tag{3.24}$$

where $H(\varphi)$ is any smooth monotone increasing function of φ with $H(0) = 0$. ENO schemes may be used to approximate the Hamilton–Jacobi equation efficiently [26, 27]. The re-initialization process also has the effect of stabilizing the high frequency noise. We will comment on this in detail in the numerical section.

3.3. *Updating the Interface by the Localized Level Set Method*

Another issue related to the level set approach is to reduce the computational cost. A natural way is to introduce a tube along the moving interface and just compute the level set function inside the tube [1, 39]. This is also the approach used in this paper.

When the traditional level set method is used to capture the moving interfaces in simulations of fluid dynamics problems, some difficulties may occur.

- It may be expensive to extend the normal velocity field, which is often only physically meaningful at the interface, to the whole domain. This is true especially when global quantities are used to determine the normal velocity at the interface or when the normal velocity field is singular. These two scenarios actually occur in our Hele–Shaw example. A simple method to extend the normal velocity off the interface can be found in [5].
- In the classical level set method, we need to calculate φ at all grid points, which involves an extra unnecessary order of magnitude of calculation.

Both these difficulties can be very well addressed by the fast localization technique introduced in [39] and details can be found there. The whole computation for the level set method is now only done in a very narrow tube around the moving interface. The size of the tube is fixed and can be just a few grid cells wide. The whole process is very simple and intuitive mathematically. It is composed of the following three steps at each time level (without loss of generality, here, we suppose that the initial level set function is a signed distance to the interface):

1. Update the level set function in a tube of width $\varepsilon_1 > 0$ by the evolution PDE for the level set function

$$\varphi_t + \mathbf{u} \cdot \nabla\varphi = 0.$$

In the tube, $\mathbf{u}$ is defined and $\mathbf{u} \cdot \mathbf{n}$ is continuous.

2. Construct a new tube of width $\varepsilon_2 > \varepsilon_1$ around the new interface (zero level set of the updated level set function) by correctly adding and deleting grid points following the moving direction of the interface.

3. Re-initialize the level set function in the new ε_2 tube by solving the nonlinear partial differential equation

$$\varphi_\tau + \operatorname{sign}(\varphi)(|\nabla\varphi| - 1) = 0 \tag{3.25}$$

to a steady state with the evolution procedure. The updated level set function will be a good approximation to the signed distance function; see [33, 38].

This localization technique involves an upwind scheme which requires only one boundary condition. However, with the tube approach, the boundary of the old tube is no longer needed. Thus an explicit tube boundary condition, which can be very difficult to prescribe and may also affect the computation, is avoided. Also any discontinuities at the edge of the tube do not affect the computation in the tube; see [39, 40]. This method works easily even in the presence of topological changes.

4. THE HYBRID METHOD FOR HELE–SHAW FLOW

In this section, we apply our hybrid immersed interface/level set method to the Hele–Shaw flow. A number of technical implementation issues will be discussed in detail.

In 1958, Saffman and Taylor [31] performed experiments replacing a viscous fluid from between two closely spaced, parallel plates with a less viscous fluid. The shape of the interface is well known to exhibit a fingering phenomenon. The velocity field $\mathbf{u} = (u, v)$ of the flow is proportional to the gradient of the pressure p. The governing equations are

$$\mathbf{u} = -\beta \nabla p, \tag{4.26}$$

$$\nabla \cdot \mathbf{u} = \phi, \tag{4.27}$$

with $\beta = b^2/(12\mu)$, where b is the gap width and μ is the viscosity, which is very different inside and outside the interface separating the two fluids. The source term ϕ is the result of the injection of the less viscous fluid into the Hele–Shaw cell. In our tests, we set

$$\phi = \begin{cases} \phi_0(r_0)(1 + \cos(r\pi/r_0)), & \text{if } r \le r_0, \\ 0, & \text{if } r > r_0, \end{cases} \tag{4.28}$$

where $r = \sqrt{x^2 + y^2}$. The total injection rate is

$$\overline{\phi} = \int\int \phi(x, y)\, dx\, dy = \phi_0(r_0)\left(\pi - \frac{4}{\pi}\right) r_0^2. \tag{4.29}$$

Specifically, if $\phi_0(r_0) = 1/r_0^2$, then we will have a single point source at the origin as r_0 approaches zero. The jump conditions across the interface are

$$[p] = \tau\kappa, \quad \text{the Laplace–Young condition,} \tag{4.30}$$

$$[\beta p_n] = 0, \quad \text{the kinematic interface condition,} \tag{4.31}$$

where τ is the surface tension and κ is the curvature of the interface.

Following the discussion of [7], we assume the less viscous fluid inside the Hele–Shaw cell has a negligible viscosity while the more viscous fluid has a finite viscosity and it is incompressible. The amalgamated surface tension[2] parameter with the dimension of length is defined

$$d_0 = \frac{2\tau\pi\beta^+}{\overline{\phi}}, \tag{4.32}$$

where $\overline{\phi}$ is the injection rate.

Linearization about a Hele–Shaw cell with radius $R(t)$ and injection rate $\overline{\phi}$ gives the instantaneous growth rate (see [7, 13]):

$$\sigma_k(t) = \frac{1}{R^2(t)}\left((k-1)\frac{\overline{\phi}}{2\pi} - \frac{d_0}{R(t)}(k-1)k(k+1)\right). \tag{4.33}$$

For a constant injection rate, we obtain a Mullins–Sekerka type instability, which shows the competition between the destabilizing effect due to the injection and the stabilizing effect due to the surface tension.

Note that, for high frequency modes, $\sigma_k(t)$ is negative, indicating that the Hele–Shaw flow is stable for these frequencies. For lower frequencies, depending on the parameters, $\sigma_k(t)$ can be positive, indicating unstable growth. Usually it is relatively easier to control high frequency noise. But it is more difficult to control the roundoff errors of low to intermediate frequencies.

The numerical simulation of the Hele–Shaw flow has attracted a lot of attention and served as a benchmark problem for numerical algorithms to compute unstable fronts [6–8, 13, 24, 29, 35, 37]. This is an ideal test model for our proposed algorithm since (4.26) can be written as

$$\nabla(\beta\nabla p) = -\phi. \tag{4.34}$$

To determine the boundary condition on the pressure, we assume that the interface is far away from the boundary so that the flow at the boundary agrees with the radial outflow which would arise from the source term in a uniform fluid; i.e.,

$$p(x, y) = p_0 - \frac{\phi_0}{2\pi\beta}\log r \tag{4.35}$$

is specified on the boundary, where p_0 is some arbitrary constant. In the numerical tests, we found that the boundary condition has little effect on the motion of the interface until the interface gets very close to the boundary.

4.1. *Computing the Velocity Field near the Interface*

When we solve the modified Hamilton–Jacobi equation to update the level set φ, we need to compute the velocity

[2] This is similar to the Atwood ratio described in [35].

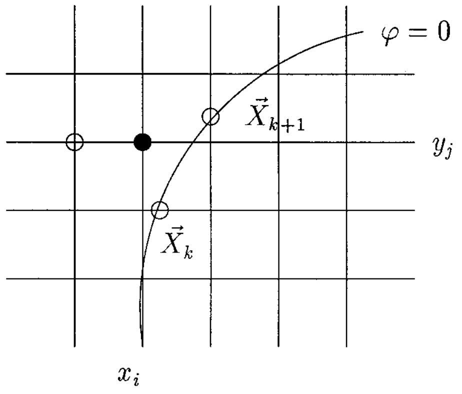

FIG. 3. Diagram of an irregular grid point and the choice of three points for interpolating the normal velocity.

field from the pressure. The velocity field can be given either in the component form in the x and y directions or $\mathbf{u} \cdot \mathbf{n}$, the normal velocity of the level set. With the fast immersed interface method, we have the normal derivatives p_n^+ and p_n^- at control points (X_k, Y_k) once the pressure p is computed. We also know the normal velocity at each control point, which is

$$u_n = -\beta p_n. \tag{4.36}$$

Note that βp_n is continuous, so it does not matter which side the quantity is taken from. If a grid point happens to be on the interface, we can use (4.36) directly. For other grid points in the small tube of the interface, we use different methods to calculate the normal velocity inside and outside the Hele–Shaw bubble. Another possible way to extend the normal velocity off the front can be found in [5].

4.1.1. *Interpolating the Normal Velocity Outside the Hele–Shaw Cell.* At a regular grid point, which the standard five-point stencil does not cut through, the normal velocity can be computed from

$$u_n = -\beta p_n = -\beta \nabla p \cdot \mathbf{n}, \tag{4.37}$$

where n is the unit normal direction and ∇p can be computed from the standard central difference scheme.

At an irregular grid point, say, (x_i, y_j) in Fig. 3, care has to be taken to compute u_n. If the grid point happens to be a control point (X_k, Y_k), then we know the normal velocity from (4.36) already.

If the irregular grid point is not a control point, the following approach is used to find the normal velocity of the level set function at this grid point:

1. Find the two closest control points $\mathbf{X}_k$ and $\mathbf{X}_{k+1}$ from the irregular grid point (x_i, y_j). We know the normal velocity at these two control points after we have solved the pressure using the immersed interface method.

2. Find a regular grid point, say (x_{i_0}, y_{j_0}), which is close to (x_i, y_j), $\mathbf{X}_k$, and $\mathbf{X}_{k+1}$ on the same side of the interface as the irregular grid point. The normal velocity at this point can be calculated by the central difference scheme using (4.37).

3. Interpolate the normal velocity at the three points to get an approximation at the irregular grid point (x_i, y_j),

$$\begin{aligned} u_n(x_i, y_j) &= \alpha_1 u_n(x_{i_0}, y_{j_0}) + \alpha_2 u_n(X_k, Y_k) \\ &\quad + \alpha_3 u_n(X_{k+1}, Y_{k+1}), \end{aligned}$$

where α_1, α_2, and α_3 is the solution of the following linear system:

$$\begin{aligned} \alpha_1 + \alpha_2 + \alpha_3 &= 1, \\ \alpha_1(x_i - x_{i_0}) + \alpha_2(x_i - X_k) + \alpha_3(x_i - X_{k+1}) &= 0, \\ \alpha_1(y_j - y_{j_0}) + \alpha_2(y_k - Y_k) + \alpha_3(y_j - Y_{k+1}) &= 0. \end{aligned}$$

This is a second-order interpolation scheme.

4.1.2. *Extending the Velocity Inside the Hele–Shaw Cell.* The algorithm described above for computing the normal velocity is successful for the irregular grid points outside of the Hele–Shaw cell, but it does not work well for those grid points inside the Hele–Shaw cell. This can be explained as follows: Compared to the solution domain, the size of the Hele–Shaw cell is very small, especially at the beginning. The potential corresponding to a single point source or a small mass of sources is

$$p(r) \sim \frac{\log(r)}{2\pi}, \quad r = \sqrt{x^2 + y^2},$$

assuming that the source centers at the origin. So the normal derivative p_n inside the Hele–Shaw bubble is nearly singular. On the other hand, from the flux condition

$$\beta^+ p_n^+ - \beta^- p_n^- = 0,$$

we also conclude that $|p_n^-|$ is very small compared to p_n^+ since $\beta^- \gg \beta^+$. Thus the normal derivative p_n changes very rapidly inside the Hele–Shaw bubble which makes the computation very difficult even at regular grid points.

The solution to this difficulty is to extend the normal velocity from the information on the interface, which is known once we have solved for the pressure. A simple extension is

$$u_n(\mathbf{x}) = \begin{cases} \dfrac{u_n(\mathbf{x}_p)}{2}\left(1 + \cos\left(\dfrac{\pi d_{\mathbf{x},\mathbf{x}_p}}{\alpha}\right)\right), & \text{if } d_{\mathbf{x},\mathbf{x}_p} \le \alpha, \\ 0, & \text{otherwise,} \end{cases} \tag{4.38}$$

where $\mathbf{x}_p$ is the projection of $\mathbf{x}$ on the interface and $d_{\mathbf{x},\mathbf{x}_p}$ is the distance between $\mathbf{x}$ and $\mathbf{x}_p$. Since the Hele–Shaw bubble is expanding outwards and we only need a few values of u_n inside a small tube containing the interface. The value α is chosen between 4 h and 6 h.

4.2. *Re-initialization Revisited*

Initially, the level set function $\varphi(x, y)$ is often chosen as the signed distance from the interface which means $|\nabla\varphi| = 1$. However, while Eq. (3.15) will move the interface (the level set $\varphi = 0$) at the correct speed, φ will, in general, no longer be a distance function. A re-initialization process is often necessary to keep φ as the signed distance function near the front within a certain accuracy, especially when the velocity field is singular at the front.

For the fast level set method in which we only update the level set in a narrow tube, there is a discontinuity along the boundary of the tube. Such a discontinuity will remain if there is no re-initialization process. The level set method would break down once the interface is close enough to the tube boundary. This is another reason we need to use a re-initialization process.

High order accurate methods are generally less stable. For interface problems such as Hele–Shaw flow, the solution is not smooth or even discontinuous. For example, the pressure in the Hele–Shaw flow is discontinuous across the interface. For these problems, straightforward discretizations may introduce high frequency numerical instabilities; see, e.g., [7, 12, 13]. This instability can be controlled by using some kind of filtering; see [13]. In our method we introduce numerical dissipation or numerical viscosity. We use a second-order essentially nonoscillatory (ENO) scheme in solving the modified Hamilton–Jacobi equation to preserve the sharp interfaces (corners or cusps). The effect of our numerical dissipation is of second order and is grid size dependent. Its effect on low frequency modes is very small. For the Hele–Shaw flow, the unstable modes are low frequencies. So our method has good resolution and accuracy when the physical viscosity is not too small. But if the physical viscosity is significantly smaller than the grid size, then our method will not be able to resolve the real physical phenomena; see also Section 5.3.

In our numerical calculations, we do not perform the re-initialization process at every time step. Instead, we carry out the re-initialization process after every 10–20 time steps using Eq. (3.25) at the grid points (i, j) in the ε_2 tube if $|\varphi(i, j)| > h$, where h is the grid size. The level set function φ is replaced by an approximation to the signed distance function in the new ε_2 tube. This process removes the stiffness at the tube boundary due to our local level set method, and it also removes high frequency noise. Such a process seems to be optimal in the sense that it will give better resolution with the least time; see Section 5.4. We also perform the re-initialization process at points in the ε_2 tube after a long time period, for example, every 100 iterations. Now the instability or oscillations caused by the high order Poisson solver (for the pressure) at the interface are eliminated by the numerical dissipation.

We have observed in our numerical experiments that for well-posed or stable problems, or for Hele–Shaw simulation with larger surface tension, it does not make much difference how often we perform the re-initialization process; see Figs. 7a, b and Figs. 9a, b. Note that the level set procedure itself imposes a "topological" regularization on unstable problems (see [10, 11]). However, for very unstable problems, which are sensitive to the parameters and roundoff errors, the different re-initialization process will affect the computational results, including the location of the interface, the area, etc. (see Figs. 7c, d, Figs. 9c, d, and the next section for more discussion). A proper choice of the re-initialization process improves our numerical results.

5. NUMERICAL EXPERIMENTS OF HELE–SHAW FLOW[3]

We have done a number of numerical experiments with different initial interfaces, viscosities, and surface tensions. All the results seem to agree with the theoretical analysis and numerical results in the literature. Since the Hele–Shaw flow is unstable for long time computations, the results do not converge to a unique solution due to the roundoff and the discretization errors. This is consistent with experiments in which different shapes are observed after some time. However, this should not invalidate our simulations because we still can predict roughly the shape and location of the interface as time evolves. Moreover, for a short time period, the solution does converge and the computational result is independent of the grid. The crucial parameter which affects the stability is the amalgamated surface tension d_0, defined in (4.32). The smaller the d_0, the more unstable the Hele–Shaw flow. Below we present some results and analysis.

[3] A short movie of the animation is available upon the request.

5.1. *A Benchmark Problem*

If the initial interface is a circle, there is only a single mode $k = 1$ in the linear stability analysis (4.33). We can construct an exact solution for this axi-symmetric flow corresponding to a known source term (see Fig. 4). We use this exact solution to check if our method works properly. Let the pressure be given

$$p(r) = \begin{cases} \dfrac{V_0}{\beta^-}\left(\dfrac{2r^3}{3\alpha^2} - \dfrac{3r^2}{2\alpha}\right) + C_1, & \text{if } 0 \le r \le \alpha, \\ -\dfrac{V_0\,\alpha}{\beta^-}\log(r) + C_0, & \text{if } \alpha < r \le r_\Gamma, \\ -\dfrac{V_0\,\alpha}{\beta^+}\log(r), & \text{if } r_\Gamma \le r, \end{cases} \tag{5.39}$$

where $r_\Gamma = \sqrt{2\alpha V_0 t + r_0^2}$ and r_0 is the radius of the initial interface. Note that the pressure is continuous across the circle $r = \alpha$. C_0 is chosen as

$$C_0 = \frac{\tau}{r_\Gamma} + V_0\alpha\log(r_\Gamma)\left(\frac{1}{\beta^-} - \frac{1}{\beta^+}\right) \tag{5.40}$$

so that $[p]_{r_\Gamma} = \tau\kappa = -\tau/r_\Gamma$. The pressure is continuous across the inner boundary $r = \alpha$ surrounding the source. The constant C_1 is chosen such that $[p]_{r=\alpha} = 0$,

$$C_1 = C_0 - \frac{V_0\alpha\log(\alpha)}{\beta^-} + \frac{5V_0\alpha}{6\beta^-}. \tag{5.41}$$

The velocity is determined from (4.26). In polar coordinates, we have

$$u_r = -\beta\frac{\partial p}{\partial r} = \begin{cases} -V_0\left(\dfrac{2r^2}{\alpha^2} - \dfrac{3r}{\alpha}\right), & \text{if } 0 \le r \le \alpha, \\ \dfrac{\alpha V_0}{r}, & \text{if } \alpha \le r, \end{cases} \tag{5.42}$$

$$u_\theta = \frac{1}{r}\frac{\partial p}{\partial \theta} = 0. \tag{5.43}$$

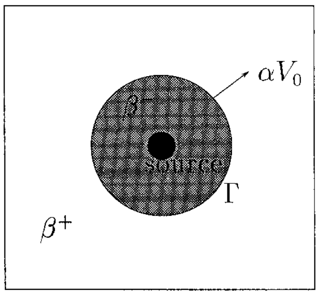

FIG. 4. An expanding interface with a constant speed αV_0.

TABLE I

Grid Refinement Analysis in the Infinity Norm, $t_0 = 0$, $t_{\text{out}} = 0.1$

n	E_p^n	Rate
40	2.4230×10^{-3}	
80	7.8189×10^{-4}	1.6318
160	1.4923×10^{-4}	2.3894
320	3.7564×10^{-5}	1.9901

Note. The parameters are: $V_0 = 0.25$, $\beta^+ = 1$, $\beta^- = 100$, $\alpha = 0.1$, $d_0 \approx 2.5 \times 10^{-3}$.

The pressure p satisfies the following Poisson equation:

$$\begin{aligned} \nabla(\beta(x,y)\nabla p) &= \frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial p}{\partial r}\right) + \frac{1}{r^2}\frac{\partial^2 p}{\partial\theta^2} \\ &= \begin{cases} V_0\left(\dfrac{6r}{\alpha^2} - \dfrac{6r}{\alpha}\right), & \text{if } 0 \le r \le \alpha, \\ 0, & \text{if } \alpha < r, \end{cases} \\ &= -\nabla\cdot\mathbf{u}. \end{aligned} \tag{5.44}$$

Thus we have the exact solution for the pressure p, the velocity field $\mathbf{u}$, and the location of the interface $\Gamma(t)$.

Note that the benchmark problem described here is not a trivial one. It is, indeed, the solution of the Hele–Shaw flow with specific source and initial interface. Although we have to specify the boundary condition, it is consistent with our discussion in the beginning of this section; see (4.35). The solutions of the pressure and the velocity depend on the surface tension, curvature, radius of the initial circle, and the injection rate.

Our numerical computations show that, for a short period of time and modest surface tension, we can obtain second-order accuracy for the pressure p, the interface location r_Γ, and the velocity vector (u, v). Table I shows the result of the grid refinement analysis for the pressure at the fixed time $t_{\text{out}} = 0.1$, where

$$E_p^n = \max_{ij}\left|p(x_i, y_j, t_{\text{out}}) - P_{ij}^n\right|,$$

$p(x_i, y_j, t_{\text{out}})$ is the exact solution at time t_{out}, P_{ij}^n is the computed solution at that time. The rate is calculated from the expression

$$\text{Rate} = \frac{\log(E_p^n/E_p^{2n})}{\log 2}.$$

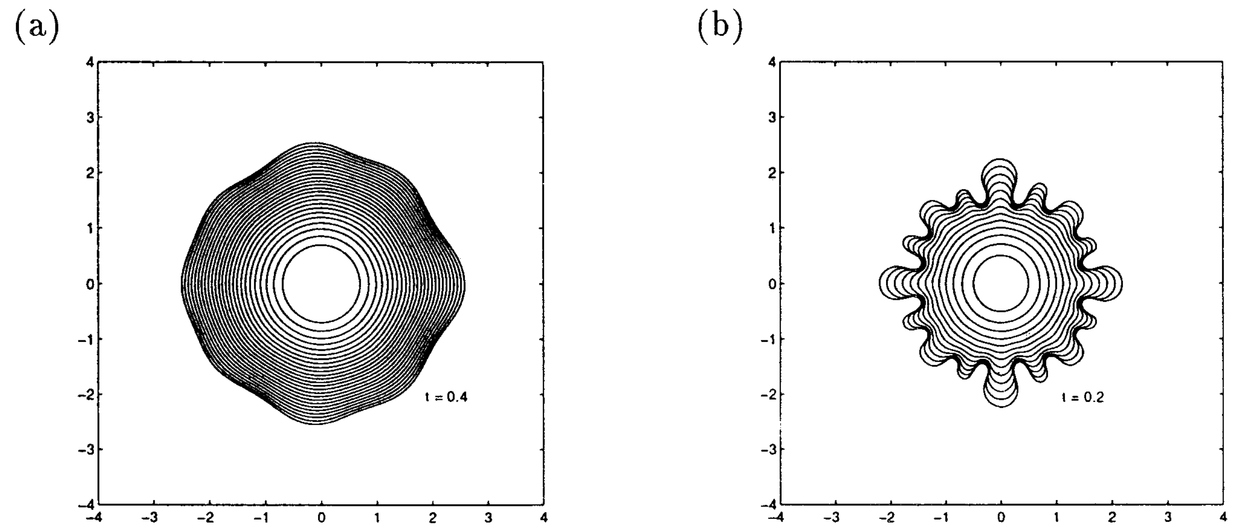

FIG. 5. Benchmark problem computation with different surface tension. The grid size is 320 by 320. (a) $d_0 = 1.256 \times 10^{-2}$, the time is from 0 to 0.4; (b) $d_0 = 2.513 \times 10^{-3}$, the time is from 0 to 0.2; more fingers are developed.

Second-order accuracy is observed. The initial interface is $r = 0.5$. The other parameters are:

$$V_0 = 0.25, \quad \beta^+ = 1, \quad \beta^- = 100, \quad \alpha = 0.1.$$

The amalgamated surface tension corresponding to the above parameters is about 2.5×10^{-3}.

For a longer time computation, there will be some low to intermediate frequency unstable modes if the surface tension is small enough to produce finger splitting. In this case, the roundoff errors associated with these modes become significant. The re-initialization process only controls the roundoff errors of high frequencies, but it does not remove effectively the roundoff errors for low to intermediate modes. So the roundoff error perturbations in the low to intermediate modes can trigger instability to this problem. Figure 5 shows the computations with different surface tensions. In Fig. 5a, the amalgamated surface tension is 1.256×10^{-2} and the flow is more stable. The circular shape is preserved for a relatively long time. In Fig. 5b, the amalgamated surface tension is 2.513×10^{-3} which is small enough to produce more fingers. In this graph, we also see some grid orientation effects in the computation. Such effects seem to be due to the source term rather than the boundary condition. In our discretization, we cannot have an exact axi-symmetric source. The source has more influence in the vertical and horizontal directions than the diagonal directions. We will see later that such an effect also exists in other figures presented in this section.

In our code, we have used the truncated GMRES(m) method to avoid running out of memory on workstations. The storage required for the GMRES(m) is of order $O(mN)$, where $N \geq m$ is the number of control points on the interface. Using the fast immersed interface method [20], the number of iterations for the Poisson equation (2.5) never exceeded 60 in our testing problems. As long as m is larger than the number of iterations, the truncated GMRES(m) method is equivalent to the full version of the GMRES method and the convergence speed is the same. We take $m \leq 160$ and $N \leq 1280$ in our simulations to get reasonably developed interfaces with available memory on the workstations. It is worthwhile to note that even with a modest workstation, our method can produce very rich structures for the Hele–Shaw problem.

5.2. *A Grid Refinement Analysis*

Now let us start with an interface which is a perturbed circle centered at the origin,

$$r_0 = 0.9 + 0.1\sin(3\theta), \quad 0 \leq \theta \leq 2\pi. \tag{5.45}$$

The flow is symmetric with respect to the y axis but not axi-symmetric. Figure 6 and Table II show the result of the grid refinement analysis with fixed amalgamated surface tension. We start with a uniform grid 80 by 80, and double it twice to conduct the grid refinement analysis. Figure 6 qualitatively demonstrates convergence of our method as we refine the mesh. Table II shows quantitatively the grid refinement analysis at three different times, against the result computed on the finest mesh size 320 by 320, since we do not know the exact solution.

The way to calculate the errors is explained below. For a closed curve, we can find its arc-length parameterization $(x(s), y(s))$ with the initial point on the x-axis, $(x(0), y(0)) = (x_0, 0)$ with $x_0 > 0$. Suppose the total arc-length of the curve is L, we divide the interval $[0, L]$ into N equally spaced subintervals, with node points being $s_i = iL/N, i = 0, 1, ..., N-1$. In this way we will have N equally spaced points $\mathbf{X}(s_i) = (x(s_i), y(s_i))$ on the closed curve. Taking $N = 320$, the errors in Table II typically is defined as

$$e_l(t) = \max_{0\le i\le N-1} \left|\mathbf{X}_l^t(s_i) - \mathbf{X}_{320}^t(s_i)\right|,$$

where $l = 80$ or $l = 160$ and $\mathbf{X}_l^t(s)$ and $\mathbf{X}_{320}^t(s)$ are the computed interfaces in the arc-length parameterizations with initial points in the positive direction of the x-axis, using the grids l by l and 320 by 320, respectively, at time t. In other words, the error is the largest distance of the corresponding points of the two computed interfaces. Since we compare the error against the solution obtained from the finest grid, 320 by 320, not the exact solution, the error ratios will also be different from the standard grid refinement analysis. If the method is second-order accurate, the ratio of e_{80}/e_{160} should be between 4 and 5. Similarly, if the method is first order, the ratio should be between 2 and 3; see [18, 19] for the details. The results in Table II clearly indicate second-order accuracy for a fixed time. On the other hand, we can also see some effects of the numerical dissipation on a coarse grid from Fig. 6.

5.3. *Further Experiments and Analysis*

Below we present some tests for the Hele–Shaw flow and compare our results with those obtained in the literature.

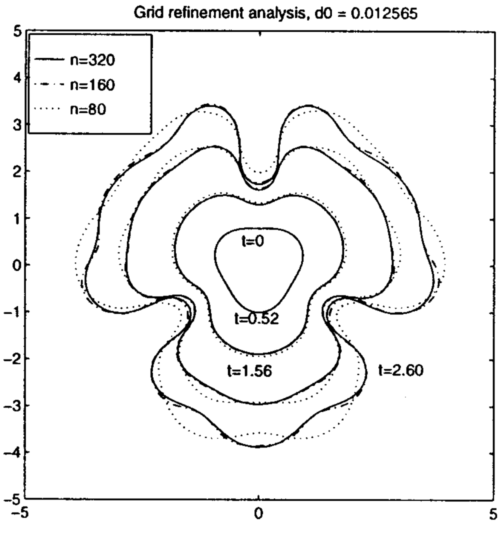

FIG. 6. A grid refinement analysis with $d_0 = 6.3 \times 10^{-3}$. Convergence is observed. On an 80 × 80 coarse grid, we observe some numerical dissipation.

TABLE II

Grid Refinement Analysis at Three Different Time, $t = 0.52$, 1.56, and 2.56 with the Amalgamated Surface Tension Being $d_0 = 6.3 \times 10^{-3}$

Time	e_{80}	e_{160}	e_{80}/e_{160}
0.52	0.088191	0.019124	4.6115
1.56	0.186631	0.042779	4.3626
2.60	0.515839	0.119723	4.3086

However, we do not intend to discuss in depth the physical meanings of the simulation here. More detailed discussion of the physical mechanism can be found, e.g., in [7, 13, 24, 35].

EXAMPLE 1. The initial interface in the polar coordinates is

$$\rho = 0.5 + 0.1\,(\sin(2\theta) + \cos(3\theta)), \quad 0 \le \theta \le 2\pi.$$

A similar example has been used in [13] with different scaling. There is no particular symmetry on the motion. Because of the surface tension, the interface remains smooth all the time. It is referred as q-pole data in [7]. Figure 7 shows the expansion of a Hele–Shaw bubble with different surface tensions at almost equally spaced time increments[4] $\Delta t \sim O(h^2)$. We have not performed a systematical time stability analysis to quantify the relationship between Δt and h. The stiffness of our time integration is less severe compared to the Lagrangian boundary integral method because we use a fixed underlying grid. Moreover, the re-initialization process plays a role of geometric regularization and stabilizing high frequency components of the solution.

The amalgamated surface tension varies from 6.3×10^{-3} to 7.5×10^{-4}. In these calculations, we choose to plot the numerical solution associated with each surface tension at a time when either the number of control points has reached 1280 or the updated interface gives a comparable total arc length. The simulation displays much of the behavior that has become known to the numerical analysts working on this subject. At early stage, three main "fjords" were developed on the interface corresponding to the three Fourier modes in the initial interface. Then the three "fjords" separated into three expanding fronts. The expanding fronts developed more fingers and petals depending on the surface tension. For large surface tension, only a few low frequencies, k between 1 and 4, are unstable for each "fjord" in Fig. 7a. As we decrease the surface

[4] Our program sometimes automatically reduces the time step when the normal velocity u_n becomes too large.

tension, more Fourier modes become unstable and we see more fingers and petals, see Figs. 7b and c. The petals expand outwards and eventually tip-split into two petals while fingers have either stopped growing or have receded and been absorbed back towards the main bulk of the bubble.

For a short time, the shape of the interface varies little for different values of surface tension. The smaller the surface tension, the quicker the secondary structure (or finger tip-splitting) develops. As we decrease the physical surface tension further, the numerical surface tension or dissipation become more apparent, indicating the limitation on the real surface tension that we can resolve. But without numerical dissipation, the interface will develop unphysical *cusps* and it will take more time to solve the Poisson equation because of the corners developed on the interface if the surface tension is very small.

The envelopes of the interface in a "fjord" are almost the same, regardless of the surface tension; see Fig. 8a. This agrees with the result in [7]. The envelopes of the interface will approach to a circle asymptotically.

Figure 8b shows the expanding Hele–Shaw cell with the initial interface in the polar coordinates,

$$\rho = 0.4 + 0.05\,(\sin(2\theta) + \cos(3\theta)), \quad 0 \leq \theta \leq 2\pi. \tag{5.46}$$

In this example, the three "fjords" are hardly seen initially. With the amalgamated surface tension $d_0 = 2.513 \times 10^{-3}$, we still see the finger tip-splitting, but we cannot see clearly the three separated fronts.

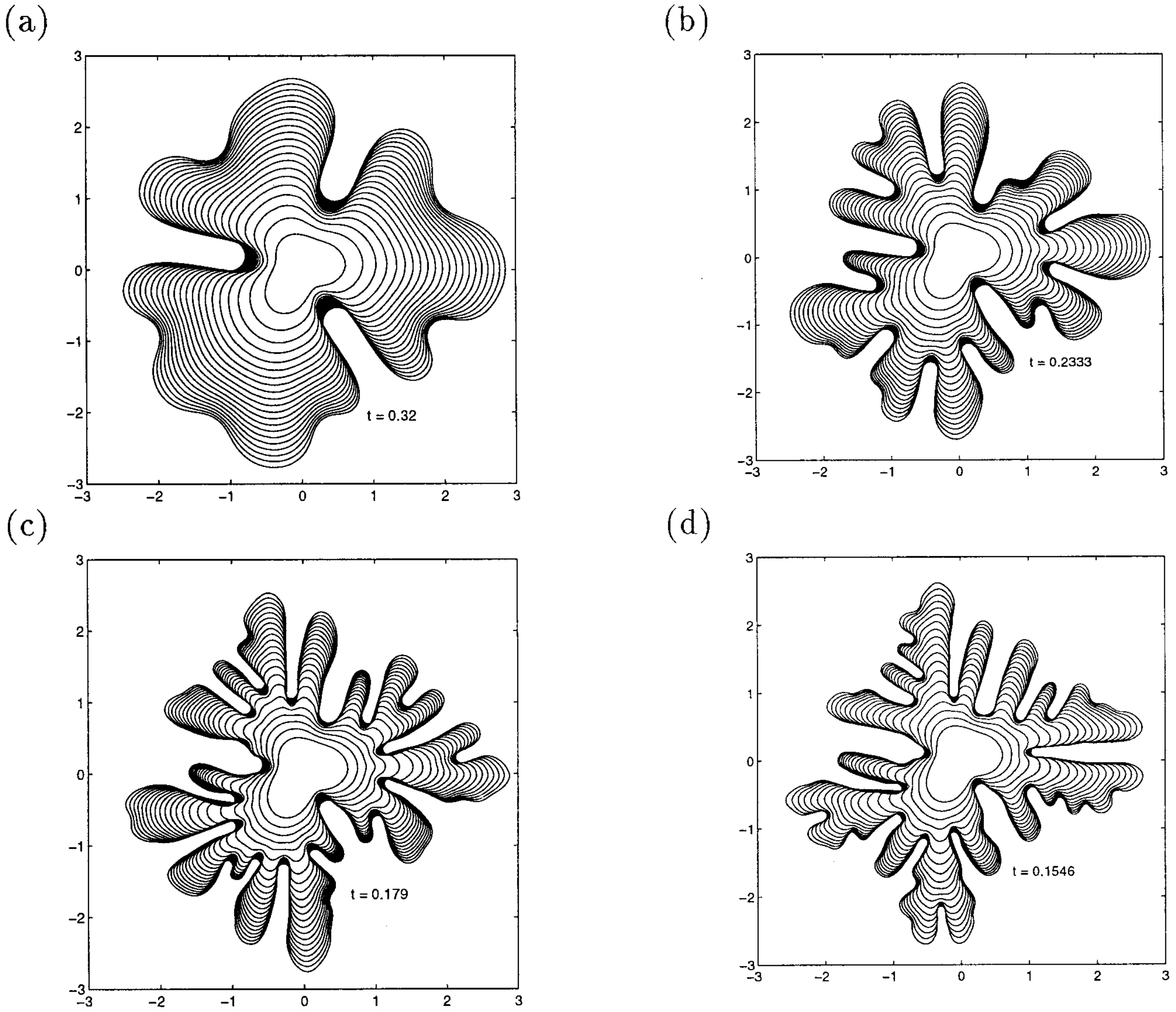

FIG. 7. Expanding Hele–Shaw bubbles with different surface tension, the grid size is 320 by 320: (a) $d_0 = 6.3 \times 10^{-3}$; (b) $d_0 = 2.513 \times 10^{-3}$; (c) $d_0 = 1.257 \times 10^{-3}$; (d) $d_0 = 7.5 \times 10^{-4}$.

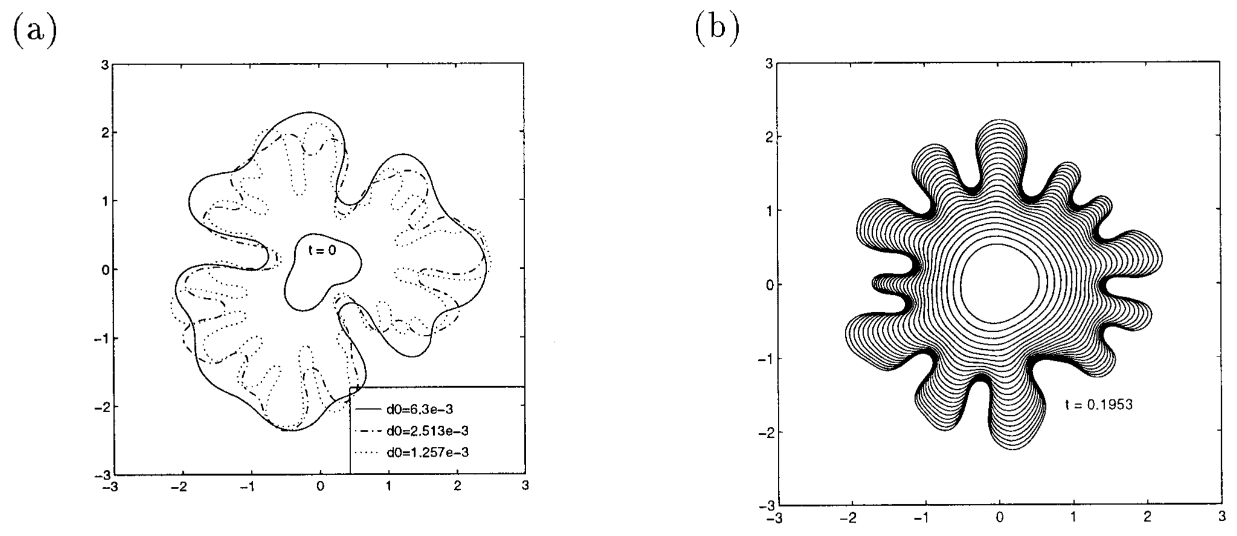

FIG. 8. (a) The snap shot of three expanding Hele–Shaw bubbles with difference surface tension. The envelopes of the interface are almost the same. (b) The initial interface is $\rho = 0.4 + 0.05\,(\sin(2\theta) + \cos(3\theta))$ with $d_0 = 2.513 \times 10^{-3}$. There is no apparent formulation of the three fjords.

Example 2. The initial interface in the polar coordinates now is

$$\rho = 0.5 + 0.1\sin(3\theta), \quad 0 \le \theta \le 2\pi.$$

It is symmetric with respect to the y axis.

Figure 9 shows computational results with different surface tension. We see pretty much the same behavior as we discussed for Example 1. The initial interface is actually axi-symmetric. After a while, we can see the effect of the roundoff errors and/or the truncation error from discretizing the source which is not exactly axi-symmetric. The source has more influence in the vertical and the horizontal directions than in the diagonal directions. The interface still roughly keeps y symmetry. The small deviation is due to the roundoff errors. Initially, only three Fourier modes are visible, the magnitude of all other modes are zero. After some time, the magnitude of some Fourier modes will grow to a significant level and appear as new fingers. There is a period of time that the magnitude of these Fourier modes is smaller than the machine precision, therefore the low frequencies of the roundoff errors will have an effect on these quantities. Since the roundoff errors are not exactly symmetric, such small perturbations will be amplified as the interface expands. We will lose the y symmetry to some extent which is visible in the pictures. Again we see the effect of numerical surface tension which determines how many unstable wave-numbers are allowed in our computation, see Fig. 9d.

Using the linear stability growth rate (4.33), we can roughly estimate how small the mesh size h should be in order to resolve all the unstable modes. The largest unstable mode k is roughly given by

$$(k-1)\frac{\overline{\phi}}{2\pi} - \frac{d_0}{R(t)}(k-1)k(k+1) \approx 0,$$

or

$$k \approx \sqrt{\frac{\overline{\phi}R(t)}{2\pi d_0}} \sim \frac{1}{\sqrt{d_0}}.$$

In order to resolve this largest unstable mode, the h should be bounded by

$$h \le \frac{1}{k} \approx \sqrt{\frac{2\pi d_0}{\overline{\phi}R(t)}} \sim \sqrt{d_0}.$$

The results presented in Fig. 7 and Fig. 9 are obtained with $h = 0.03125$ or $h^2 = 9.76 \times 10^{-4}$. That explains why we may not be able to fully resolve all the unstable modes in Fig. 7d and Fig. 9d, where $d_0 = 7.5 \times 10^{-4}$.

Example 3. This example shows a fully developed Hele–Shaw cell. The interface initially is

$$\rho = 0.2 + 0.05\sin(3\theta), \quad 0 \le \theta \le 2\pi. \tag{5.47}$$

Figure 10 shows the fully developed Hele–Shaw bubble corresponding to the initial interface. We can see that many fingers and petals are developed with time.

5.4. *Time Allocation Analysis*

We have done a lot of other tests with different parameters. Here we only present a few typical results. Most of the computations can be done within several hours on an IBM RS6000 machine with multiple users. It is hard to compare the accuracy of our method with other methods in the literature because of the different settings. But we have shown that our method is second-order accurate for fixed time. Thus its accuracy is competitive with other methods. Using a suitable flag in most Fortran compliers, we have been able to find out the time allocation for each subroutine and function call. Below we present the time allocation summary for Example 1 with a 160 by 160 grid. The amalgamated surface tension was $d_0 = 6.3 \times 10^{-3}$ and the final time was $t = 0.25$. We consider the following cases:

Case A. The original level set method which updates the level set function in the entire domain. The re-initialization process was carried out at each time level.

Case B. The tube width in the fast level method was taken as $15h$ with the re-initialization process done at each time step.

Case C. The tube width was taken as $3h$ with the re-initialization process done at each time step.

Case D. The tube width was taken as $15h$ with the re-initialization process done every 15 steps.

Table III gives a time allocation for the different cases. The same re-initialization code is used for all cases. The time needed for solving PDEs includes the fast immersed interface method for solving the Poisson equation with

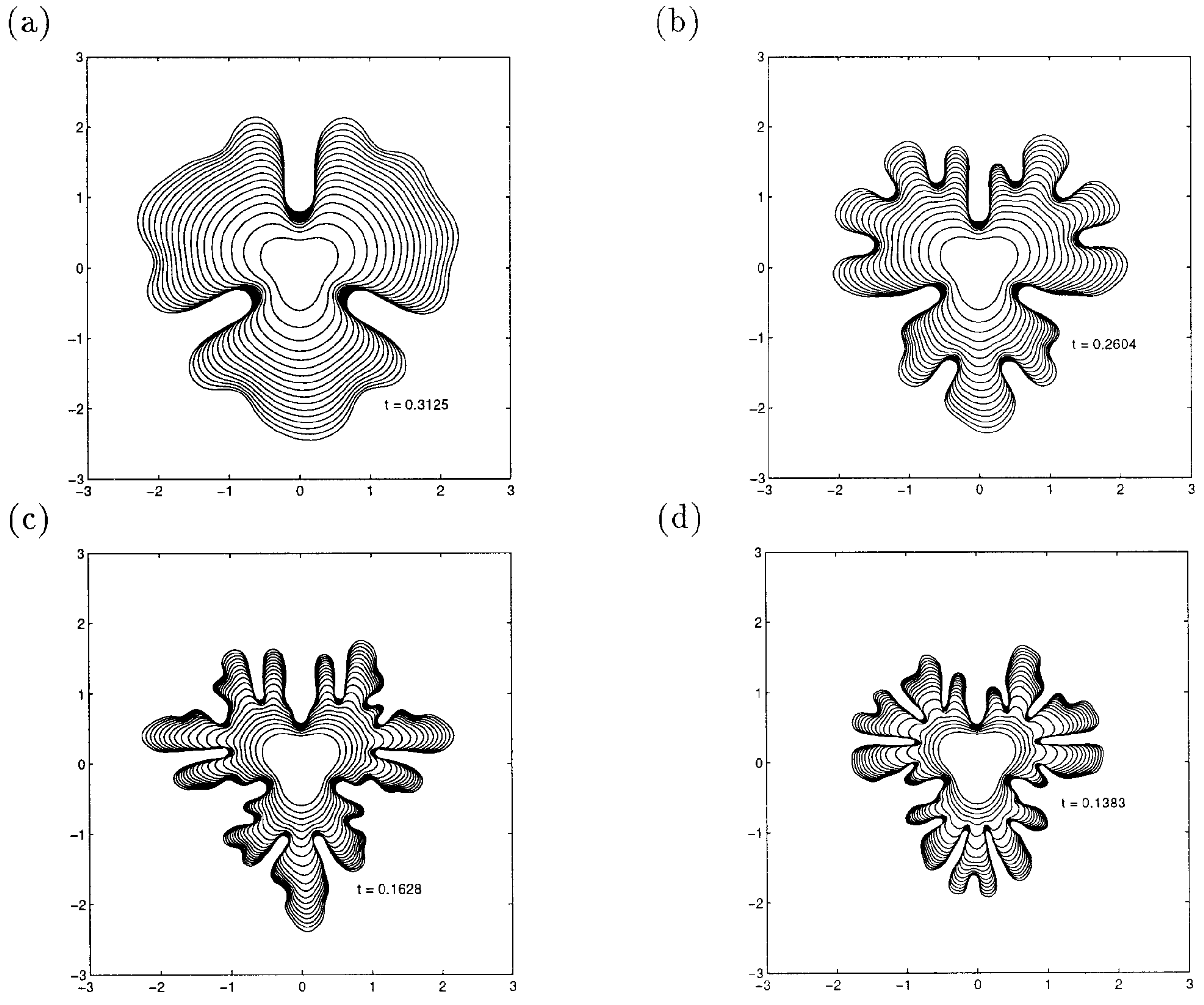

FIG. 9. Expanding Hele–Shaw bubbles with different surface tension from a y-symmetric initial bubble. The grid size is 320 by 320: (a) $d_0 = 6.3 \times 10_{-3}$; (b) $d_0 = 2.513 \times 10^{-3}$; (c) $d_0 = 1.257 \times 10^{-3}$; (d)$_0 = 7.5 \times 10^{-4}$.

discontinuous coefficients and the jump conditions (4.30) and (4.31), the interpolation scheme for the velocity field on the interface. The fast Poisson solver used is the subroutine HWSCRT from FISH pack. It is a little bit slower than the FFT method. The time needed for the level set function includes extending the normal velocity to the tube needed for the fast level set method and updating the level set function for the next time level. The time used by the system calls includes the calls of built-in functions and system-dependent routines. Our code was not optimal and the percentage of the time allocation may not be exactly correct since some routines were shared by several major components. However, Table III should give us a quantitative picture of the time allocation of major components in our method.

Intuitively, it seems that a boundary integral method which updates one-dimensional interfaces would be faster than the level set approach which updates two-dimensional level set functions. From Table III we can clearly see that, even with the original level set method and the re-initialization process at every time step, the time needed to update the interface is just a small portion of the entire computation which is less than 15%. Usually the time needed for the re-initialization process is more than that of updating the level set function since the re-initialization process contains several iterations of the level set method. With the fast level set approach, the time needed for the level set approach and the re-initialization process decreases. In most of our computations, we have used a modest tube size, say $15h$, so it is safe to do the re-initialization after every 10–15 time steps. The time needed for both processes was only about 2.7%, which is more than five times faster than the original level set method with re-initialization at every time step.

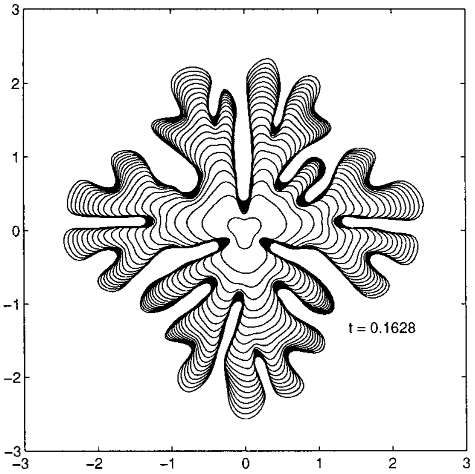

FIG. 10. An expanding Hele–Shaw bubble with initial interface: $\rho = 0.2 + 0.05\sin(3\theta)$, $0 \le \theta \le 2\pi$, $d_0 = 1.257 \times 10^{-3}$. The grid size is 320 by 320.

TABLE III

Time Allocation in Percentage for Different Cases for Example 1 with 160 by 160 Grid and the Amalgamated Surface Tension $d_0 = 6.3 \times 10^{-3}$

Case	Solve PDE	Level set	Re-initialization	System calls	Other
A	69.9	5.5	9	13.6	2
B	73.6	3.3	9	13.3	0.8
C	79	2.2	6.9	10.8	1.1
D	82.4	2.2	0.5	14.4	0.5

Note. The final time is $t = 0.25$.

The advantage of the boundary integral method is that it can be made arbitrarily high-order accurate and is free of numerical dissipation or dispersion. This property is very important when we try to study singularity formation of the interface as well as the nature of the singularity, see, e.g., [12–14]. This is especially evident when surface tension is small and new topological singularities may develop. A grid-based method would require a sophisticated adaptive grid refinement in order to produce the necessary accuracy near the singularity. On the other hand, if the purpose of the study is to obtain some qualitative understanding of the interface structure and their dynamical properties, a grid-based method can prove to be more robust. The main cost in the boundary integral method is to evaluate the velocity integral at each time step, which is an order $O(N^2)$ operations by direct summation methods, where N is the number of grid points along the interface. This difficulty can be greatly reduced by using the fast multi-pole summation method, see [13]. In this case, the operation count is reduced to $O(N)$. However, from the practical consideration, the constant in $O(N)$ is still quite large.

It is also interesting to consider the vortex sheet method [24, 35], which is based on the vorticity variables and discrete circular arcs of the interface and for which the main cost is to update the location of vortices and update the circulation until the velocities converge. In our method, the corresponding computation is to solve the Poisson equation with piece-wise constant coefficients using the fast immersed interface method. This is probably the main saving of the entire algorithms. From the time allocation analysis we also find out that the effort needed near the interface for the immersed interface method is only about 25–30% of the fast Poisson solver for a 160 by 160 grid. The ratio will decrease as we increase the number of grid points. This explains why the immersed interface method offers an efficient approach for interface problems.

6. CONCLUSIONS

In this paper, we present a new numerical method which combines the immersed interface method with the level set formulation for moving interface problems. Several ingredients of our methods such as the reconstruction of the interface, extending the normal velocity, and the bilinear interpolation to restrict a grid function to a lower dimensional space, can be applied to other problems as well. Our method is second-order accurate unless the interface develops singularities. Numerical experiments for Hele–Shaw flow demonstrate the efficiency of this method. Although there is a limitation on the problems that can be resolved by this method due to numerical dissipation, we can see a lot of applications for well-posed and stable moving interface problems, especially those involving topological changes. Even for ill-posed and unstable problems, our method can be quite robust as we have demonstrated for the Hele–Shaw flow.

ACKNOWLEDGMENTS

We thank Randall J. LeVeque for very beneficial discussions and the treatment of the boundary condition, for reading, correcting and commenting on this paper. We also thank the referees for their suggestions on the final version of the paper.

REFERENCES

1. D. Adalsteinsson and J. Sethian, *A Fast Level Set Method for Propagating Interfaces,* Lawrence Berkeley Lab Report LBL-34893, 1993.
2. L. M. Adams, A multigrid algorithm for immersed interface problems, in *Proceedings, Copper Mountain Multigrid Conference, 1995.*
3. J. Brackbill, D. Kothe, and C. Zemach, A continuum method for modeling surface tension, *J. Comput. Phys.* **100,** 335 (1992).
4. Y. C. Chang, T. Y. Hou, B. Merriman, and S. Osher, A level set formulation of Eulerian interface capturing method for incompressible fluid flows, *J. Comput. Phys.* **124,** 449 (1996).
5. S. Chen, B. Merriman, P. Smereka, and S. Osher. *A Fast Level Set Based Algorithm for Stefan Problems.* UCLA CAM report 96-21, also submitted to J. Comput. Phys., 1996.
6. W. Dai, L. Kadanoff, and S. Zhou, Interface dynamics and the motion of complex singularities, *Phys. Rev. A* **43,** 6672 (1991).
7. W. Dai and M. J. Shelley, A numerical study of the effect of surface tension and noise on an expanding Hele–Shaw bubble, *Phys. Fluids. A* **5**(9), 2131 (1993).
8. A. J. Degregoria and L. W. Schwartz, A boundary-integral method for two-phase displacement in Hele-Shaw cells, *J. Fluid Mech.,* **164,** 383 (1986).
9. L. Greengard and V. Rokhlin. A fast algorithm for particle summations, *J. Comput. Phys.,* **73,** 325 (1987).
10. E. Harabetian and S. Osher. Regularization of ill-posed problems via level set approach, *SIAM J. Appl. Math.* [submitted].
11. E. Harabetian, S. Osher, and C. W. Shu. An Eulerian approach for vortex motion, *J. Comput. Phys.,* **127,** 15 (1996).
12. T. Hou. Numerical solutions to free boundary problems, *Acta Numer.,* pp. 335–415 (1995).
13. T. Y. Hou, J. S. Lowengrub, and M. J. Shelley. Removing the stiffness from interfacial flows with surface tension, *J. Comput. Phys.,* **114,** 312 (1994).
14. R. Krasny, A study of singularity formation in a vortex sheet by the point vortex approximation, *J. Fluid Mech.,* **167,** 65 (1986).
15. R. J. LeVeque and Z. Li. The immersed interface method for elliptic equations with discontinuous coefficients and singular sources, *SIAM J. Numer. Anal.,* **31,** 1019 (1994).
16. R. J. LeVeque and Z. Li. Simulation of bubbles in creeping flow using the immersed interface method, in *Proc. Sixth International Symposium on Computational Fluid Dynamics, 1995.* pp. 688–693.
17. R. J. LeVeque and C. Zhang. Immersed interface methods for wave equations with discontinuous coefficients, wave motion, [in press]
18. R. J. LeVeque and Z. Li. Immersed interface method for stokes flow with elastic boundaries or surface tension, *SIAM J. Sci. Stat. Comput.* [in press]
19. Z. Li. *The Immersed Interface Method—A Numerical Approach for Partial Differential Equations with Interfaces,* Ph.D. thesis, University of Washington, 1994.
20. Z. Li. A fast iterative algorithm for elliptic interface problems, *SIAM J. Numer. Anal,* [to appear]
21. Z. Li. Immersed interface method for moving interface problems. *Numerical Algorithms,* [to appear]
22. Z. Li. A note on immersed interface methods for three dimensional elliptic equations. *Computers Math. Appl.,* **31,** 9 (1996).
23. Z. Li and A. Mayo, ADI methods for heat equations with discontinuities along an arbitrary interface, in *Proc. Symp. Appl. Math.,* Vol. 48, p. 311, edited by W. Gautshi, Am. Math. Soc., Providence, RI, 1993.
24. E. Meiburg and G. Homsy, Nonlinear unstable viscous fingers in Hele-Shaw flows. II. numerical simulation, *Phys. Fluids,* **31**(3), 429 (1988).
25. K. W. Morton and D. F. Mayers, *Numerical Solution of Partial Differential Equations,* Cambridge Univ. Press, Cambridge, 1995.
26. S. Osher and J. A. Sethian. Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations, *J. Comput. Phys.,* **79,** 12 (1988).
27. S. Osher and C.-W. Shu. High-order essentially nonoscillatory schemes for Hamilton–Jacobi equations. *SIAM J. Numer. Anal.,* **28,** 907 (1991).
28. C. S. Peskin. Numerical analysis of blood flow in the heart, *J. Comput. Phys.,* **25,** 220 (1977).
29. D. A. Reinelt. Interface conditions for two-phase displacement in Hele–Shaw cells, *J. Fluid Mech.,* **183,** 219 (1987).
30. Y. Saad. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems, *SIAM J. Sci. Stat. Comput.,* **7,** 856 (1986).
31. P. G. Saffman and G. I. Taylor. The penetration of a fluid into a porous medium or Hele–Shaw cell containing a more viscous liquid. *Proc. R. Soc. Lond. A,* **245,** 312 (1958).
32. M. Sussman, Ph.D thesis, UCLA CAM Report 94-13, 1994.
33. M. Sussman, P. Smereka, and S. Osher. A level set approach for computing solutions to incompressible two-phase flow, *J. Comput. Phys.,* **114,** 1994.
34. Paul N. Swarztrauber. *Fast Poisson Solver,* in *Studies in Numerical Analysis,* edited by G. H. Golub, Vol. 24, p. 319 Mattr. Assoc. Am., Washington, DC, 1984.
35. G. Tryggvason and H. Aref. Numerical experiments on Hele-Shaw flow with a sharp interface, *J. Fluid Mech.,* **136,** 20 (1983).
36. S. O. Unverdi and G. Tryggvason. A front-tracking method for viscous, incompressible, multifluid flows, *J. Comput. Phys.,* **100,** 25 (1992).
37. N. Whitaker. Some numerical methods for the Hele-Shaw equations, *J. Comput. Phys.,* **111,** 81 (1994).
38. H. Zhao, T. Chan, B. Merriman, and S. Osher. A variational level set approach to multiphase motion, *J. Comput. Phys.,* **127,** 179 (1996).
39. H. Zhao, M. Kang, B. Merriman, J. Shao, and S. Osher, A localized and PDE based level set method, preprint.
40. H. Zhao, J.-P. Shao, S. Osher, T. Chan, and B. Merriman. Motion of multiple junctions and interfaces in 3-D and applications to domain decomposition, preprint, 1995.