Journal of Computational Physics **171,** 243–263 (2001)
doi:10.1006/jcph.2001.6785, available online at http://www.idealibrary.com on IDEAL®

Numerical Simulation of Moving Contact Line Problems Using a Volume-of-Fluid Method

Michael Renardy,* Yuriko Renardy,* and Jie Li†

**Department of Mathematics, Virginia Tech, Blacksburg, Virginia 24061-0123; †BP Institute, Bullard Labs, University of Cambridge, Cambridge CB3 0EZ, United Kingdom*
E-mail: renardym@math.vt.edu

Received March 28, 2000; revised January 22, 2001

Moving contact lines are implemented in a volume-of-fluid scheme with piecewise linear interface construction. Interfacial tension is treated as a continuous body force, computed from numerical derivatives of a smoothed volume-of-fluid function. Two methods for implementing the contact angle condition are investigated. The first extrapolates the volume-of-fluid function beyond the flow domain, on the basis of the condition that its gradient is perpendicular to the interface and that the normal to the interface at the wall is determined by the contact angle. The second method treats the problem as a three-phase situation and mimics the classical argument of Young. It is found that the latter approach introduces an artificial localized flow, and the extrapolation method is preferable. Slip is a crucial factor in the spreading of contact lines; the numerical method introduces slip at the discrete level, effectively introducing a slip length on the order of the mesh size. © 2001 Academic Press

Key Words: contact line; volume of fluid method.

1. INTRODUCTION

Flows involving fluid interfaces arise in a multitude of applications, including water waves, lubricated pipeline transport, manufacturing of multilayer films, and fibers [9]. The principal question in many of these flows is how the two fluids arrange themselves, and there are many situations where the topology of the fluid interface changes during the flow. The volume-of-fluid (VOF) method has become a preferred choice to deal with such situations, because it requires no *a priori* assumptions on the nature of the fluid interface, and, therefore, no special procedures are needed to handle changes in the topology of the interface, for instance, in the breakup of drops [14].

Over the past few years, the authors have developed the code SURFER++, which expands on the SURFER code of Li and Zaleski [11]. The method employs a VOF scheme on a MAC finite difference grid, with a continuous surface stress algorithm for the surface tension force. We refer to [11]–[14] and to the following section of this paper for specifics. Until now, however, the code did not allow for contact lines. Contact lines are an essential part of many applications involving interfacial evolution. An example is the encapsulation phenomenon in pipe flows. Contact lines have been implemented in earlier VOF codes (see e.g. [10]), but we are not aware of a systematic evaluation of these codes in situations involving contact lines. In this article, we present a formulation of the numerical scheme together with performance tests. The main features are elucidated for the two-dimensional case, and the extension to three dimensions is straightforward. The test problems have been chosen as simply as possible while at the same time highlighting the performance of the essential component parts of the code.

The VOF method is based on a "volume-of-fluid function" which gives the volume fraction of one of the fluids in each cell of the finite difference grid. Interfacial tension forces are computed as a body force, using numerical derivatives of the volume-of-fluid function (or a "smoothed" volume-of-fluid function). To compute surface tension forces near the boundary, we need to define the volume-of-fluid function outside the flow domain. We shall explore two methods for doing this. The first of these is an extrapolation method, which is based on the condition that the gradient of the volume-of-fluid function is normal to the interface, and the interface normal at the boundary is determined by the prescribed contact angle. By using this condition, we can extrapolate the volume-of-fluid function beyond the flow domain. This is a generalization of the approach that is described in [10] for a lower order version of the method. The second method is based on treating the contact region as a three-phase problem, where each of the fluids and the solids have a volume-of-fluid function associated with them. We can then compute a surface tension force from each of the three volume-of-fluid functions, and the contact angle is determined by the equilibrium of the surface tension forces on the three interfaces as in the classical argument of Young.

As is well-known, an infinite force is required to cause a contact line to move. This is a paradox, since the force provided by surface tension is finite and, on the other hand, it is believed to be the force which is responsible for moving the contact line. In considering moving contact line problems, it is therefore important to include slip (see for instance, [3, 4, 7, 8, 15, 18]). Moreover, the observed contact angle in a moving contact line is generally not equal to the static contact angle; in addition, the macroscopic contact angle is different from the actual contact angle at the wall, and determining the relationship is a complicated problem of matched asymptotics. The magnitude of slip can be quantified as a length scale, which is known as the slip length. Results in the literature [3, 4, 7, 8, 15] indicate that the magnitude of this slip length is the most essential factor in contact line spreading, while the details of the slip law have little effect on the overall flow. In numerical simulations, the physics of the problem is only resolved to the scale of the mesh, and we can expect an effective slip length on the order of the mesh size.

In Section 2, the governing equations are detailed, and novel aspects of our numerical method are described. An unforeseen issue was that mass conservation became a major problem. The version of the PLIC advection scheme for the interface used in the previous version of our code, as described in [12], is not mass conserving. This caused no problems in our earlier simulations, where errors in mass conservation were minor. However, the singular velocity field at contact lines leads to much larger errors which have a major effect on mass conservation, and we were forced to modify the scheme to make it mass conserving in order to avoid unacceptable errors. The second major change in the code is, of course, the treatment of the surface tension force near the boundary.

Section 3 describes the results of our computations. All our calculations are in two dimensions. We first consider a model problem for a circular drop on a flat surface. The initial contact angle is not the correct one, causing the drop to spread. We compare our two methods of implementing the contact angle and conclude that the extrapolation method is preferable over the three-phase method, because the latter method causes a strong localized flow near the contact point which persists even after equilibrium is reached. We also assess the effect of slip on the spreading of the contact line. Explicitly imposed slip is found to increase the speed of contact line spreading, but even in the absence of explicit slip, there is effective slip on the scale of the mesh. We find a contact line speed which is roughly inversely proportional to the logarithm of the mesh size. This is in qualitative agreement with the asymptotic study of [8], which predicts a contact line speed of the form $1/(a - b \ln \lambda)$, where λ is a slip length, and a and b depend on the drop radius and on the difference between the actual and the equilibrium contact angle. In our computations, the mesh size effectively takes over the role of the slip length. “Correct” results would therefore require a mesh refinement down to the level of the true slip length; clearly this requires adaptive meshes, which our code does not currently have. We also consider the effect of inertia and the effect of an imposed shear flow on the deformation of the drop.

The last problem we consider is that of a rising drop approaching the top. We might expect the drop to “hit” the top wall and then spread. Indeed, some prior studies of the problem [5, 6] start with an initial condition where a spherical drop is already touching the wall. For our computations, the Reynolds number is considerably lower than in [5, 6]. We find that the drop flattens out as it approaches the top wall and the point closest to the wall is actually not in the center (this behavior is also observed in [16]). Moreover, the impact occurs at a time which increases with mesh refinement. We interpret this to mean that the drop actually does not reach the wall until microscopic effects, e.g. van der Waals forces, become significant.

2. NUMERICAL METHOD

2.1. *The Equations of Motion*

The two-fluid flow is modeled with the Navier–Stokes equation

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u}\right) = -\nabla p + \nabla\cdot(\mu\mathbf{S}) + \mathrm{F}, \tag{1}$$

where ρ is the density, μ the viscosity, $\mathbf{S}$ the rate of strain tensor

$$\mathbf{S}_{ij} = \frac{1}{2}\left(\frac{\partial u_j}{\partial x_i} + \frac{\partial u_i}{\partial x_j}\right),$$

and $\mathbf{F}$ the source term for the momentum equation. In our calculations, the body force $\mathbf{F}$ includes the gravity and interfacial tension force. The velocity field $\mathbf{u}$ is subject to the incompressibility constraint

$$\nabla\cdot\mathbf{u} = 0. \tag{2}$$

The two fluids are immiscible, and named fluids 1 and 2. Density and viscosity are constant in each phase but may be discontinuous at the interface. We use a volume fraction field C to represent and track the interface which is transported by the velocity field $\mathbf{u}$:

$$\frac{\partial C}{\partial t} + \mathbf{u} \cdot \nabla C = 0. \tag{3}$$

This equation allows for the calculation of density and viscosity. In fact, the average values of density and viscosity are interpolated by the formulas

$$\rho = C\rho_1 + (1 - C)\rho_2, \tag{4}$$

$$\mu = C\mu_1 + (1 - C)\mu_2. \tag{5}$$

2.2. *Numerical Discretization*

The basic scheme of the numerical discretization is a finite difference method, in which the computational domain is meshed by a rectangular grid. Nodal values of the volume fraction C and the pressure p are defined at the center of each rectangle, nodal values of the horizontal velocity are defined at the midpoints of the vertical sides, and nodal values of the vertical velocity are defined at the midpoints of the horizontal sides.

At each time step, the algorithm consists of four basic components:

1. The interface position, i.e., the volume fraction C, is updated.
2. A surface tension force is computed.
3. A projection method is used for the Navier–Stokes solver. An intermediate velocity is computed by using the momentum equation without the pressure term. This intermediate velocity is not divergence free.
4. The pressure field is computed to yield a divergence free velocity field.

It is primarily in the first two steps where we had to make changes to the algorithm to accommodate moving contact lines.

2.3. *Updating the Interface Position*

The updating of the interface is based on piecewise linear interface construction (PLIC). The prior version of the PLIC scheme works as follows: in each rectangular cell in which C is not equal to 0 or 1, compute a direction normal to the interface using the gradient of C via $\mathbf{n} = \nabla C/|\nabla C|$. Then the interface within the cell is given by a straight line which has normal vector $\mathbf{n}$ and divides the cell according to the volume fraction C. Thus the domain occupied by each fluid within the cell is a polygon. We now move this polygon by moving its corners by the amount of velocity times time step Δt. For this purpose, we determine velocities at the corners of each rectangular cell by linear interpolation from the nodal values, and we determine velocities at the points where the interface intersects the cell boundary by linear interpolation from these corner values. By computing the areas of the intersection of the updated polygon with the original cell and with neighboring cells, we can determine how much fluid has moved into each neighboring cell. These values are then used to update the color function. To simplify programming, separate updates were used for the motion in the x-direction and in the y-direction.

The first modification we make to this scheme is to change the definition of the interface normal in cells adjacent to the boundary: instead of determining $\mathbf{n}$ from the gradient of C, we determine it from the prescribed contact angle.

A larger and more crucial modification results from the issue of mass conservation. As described above, the PLIC scheme does not conserve mass. In prior computations without contact lines, the errors in mass conservation were small, but, as computations reported below will show, the rapidly varying velocities near the singularity at the moving contact line lead to errors which, although they do tend to zero with mesh refinement, are unacceptably large. This caused us to upgrade the PLIC scheme to PLIC+, in a fashion which enforces mass conservation.

First, we change the advection scheme to conserve the area of each cell. In order to achieve this, we must change the way in which corner velocities are interpolated. Let (i, j) denote the index of a cell. Then the nodal values of the horizontal velocity are $u_{i+1/2,j}$ at the midpoint of the right boundary and $u_{i-1/2,j}$ at the midpoint of the left boundary. In the original PLIC scheme, the velocities at the upper right and lower right corner are determined as follows:

$$\begin{aligned} u_{i+1/2,j+1/2} &= \frac{1}{2}\left(u_{i+1/2,j+1} + u_{i+1/2,j}\right), \\ u_{i+1/2,j-1/2} &= \frac{1}{2}\left(u_{i+1/2,j-1} + u_{i+1/2,j}\right). \end{aligned} \tag{6}$$

If we now move the corners of the cell, the volume of the cell changes by

$$\frac{1}{4}\Delta t\,\Delta y\left(u_{i+1/2,j+1} + 2u_{i+1/2,j} + u_{i+1/2,j-1} - u_{i-1/2,j+1} - 2u_{i-1/2,j} + u_{i-1/2,j-1}\right). \tag{7}$$

We changed (6) to

$$\begin{aligned} u_{i+1/2,j+1/2} &= \frac{1}{4}\left(u_{i+1/2,j+1} + 4u_{i+1/2,j} - u_{i+1/2,j-1}\right), \\ u_{i+1/2,j-1/2} &= \frac{1}{4}\left(u_{i+1/2,j-1} + 4u_{i+1/2,j} - u_{i+1/2,j+1}\right). \end{aligned} \tag{8}$$

This is accurate to the same order, but the effect is that the change in the volume of the cell is now

$$\Delta t\,\Delta y\left(u_{i+1/2,j} - u_{i-1/2,j}\right). \tag{9}$$

We follow the analogous procedure for advection in the y-direction, the change in the volume of the cell is

$$\Delta t\,\Delta x\left(v_{i,j+1/2} - v_{i,j-1/2}\right). \tag{10}$$

We note that the discrete version of the divergence condition is precisely that the sum of the terms (9) and (10) is zero. In our advection scheme, we first compute an update C_x, which is obtained by advecting only in the x-direction and then an update C_y, which is obtained by updating only in the y-direction. The new value of C is then found as $C_x + C_y - C$.

Although the scheme now conserves the volume of each cell as a whole, it still need not conserve separately the volumes of fluid 1 and 2. This is because the velocity at points

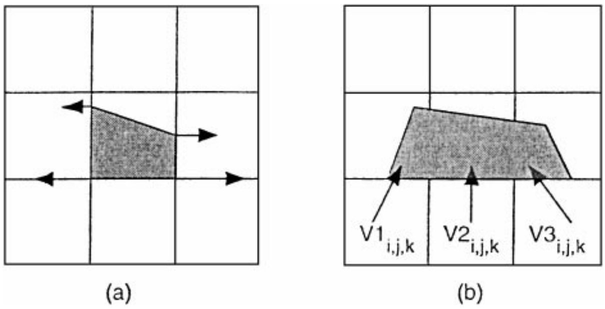

FIG. 1. Lagrangian method during advection step. (a) Shaded polygon represents part occupied by the fluid in the central cell. (b) Contribution of this part to new volume fraction field after advection.

where the interface intersects cell boundaries is still found by linear interpolation, and no divergence constraint has been imposed on these interpolated values. We force conservation by following the following procedure:

1. First, we advect in the x-direction. If the Courant stability condition is satisfied, fluid from cell (i, j) will end up in cells $(i-1, j)$, (i, j) and $(i+1, j)$ (see Fig. 1). Let $\mathsf{VOF1}$, $\mathsf{VOF2}$, and $\mathsf{VOF3}$ be the amounts of fluid 1 in cells $(i-1, j)$, (i, j) and $(i+1, j)$, respectively (measured in units of the cell size $\Delta x \Delta y$). Since the x component of the velocity by itself is not divergence free, it is generally not true that $\mathsf{VOF1} + \mathsf{VOF2} + \mathsf{VOF3} = c(i, j)$. Let EFAC be the "expansion factor"

$$\mathsf{EFAC} = (\mathsf{VOF1} + \mathsf{VOF2} + \mathsf{VOF3})/c(i, j). \tag{11}$$

2. Next, we advect in the y-direction. Let $\mathsf{VOF4}$, $\mathsf{VOF5}$, and $\mathsf{VOF6}$ be the amounts of fluid 1 from cell (i, j) which end up in cells $(i, j-1)$, (i, j) and $(i, j+1)$, respectively. We enforce mass conservation by adjusting $\mathsf{VOF4}$, $\mathsf{VOF5}$, and $\mathsf{VOF6}$ to new values $\mathsf{VOF4}'$, $\mathsf{VOF5}'$, and $\mathsf{VOF6}'$ such that

$$\mathsf{VOF4}' + \mathsf{VOF5}' + \mathsf{VOF6}' = c(i, j) * (2 - \mathsf{EFAC}). \tag{12}$$

Ordinarily, this is done by adjusting each of the values $\mathsf{VOF4}$, $\mathsf{VOF5}$, and $\mathsf{VOF6}$ by the same multiplicative factor. An exception is provided, however, if this would make the amount of fluid 1 transported from one cell to the next larger than the total volume-of-fluid transported between the same cells. If, for instance, $\mathsf{VOF4}'$ exceeds this limit, we do not adjust $\mathsf{VOF4}$ and distribute the excess only among $\mathsf{VOF5}$ and $\mathsf{VOF6}$.

While the new PLIC+ scheme enforces mass conservation, a drawback is that the values of the VOF function are no longer generated by the motion of a smooth interface (the correction applied in step 2 above is not described in terms of interface motion). We believe that this aspect of the algorithm is responsible for the production of small "flotsam and jetsam" (cells in which C is almost one but not exactly one) which is apparent in some of our figures. We found that this improved with mesh refinement and, in any case, did not have any significant effect on the rest of the computations.

2.4. *Continuous Surface Force*

The code approximates surface tension by a continuous body force, which is given as the divergence of a surface stress [1, 17]:

$$\mathbf{T}_S = (\mathbf{I} - \mathbf{n}\mathbf{n}^T)\sigma|\nabla C|, \quad \mathbf{n} = \nabla C/|\nabla C|. \tag{13}$$

To implement this formulation, the volume fraction C in this formula is replaced by a smoothed function obtained by convolution with a mollifier. To calculate the smoothed volume fraction and to evaluate the gradient of C near the boundary, it is necessary to define values of C outside the computational region. We have tested two different procedures to do this.

The first procedure is based on extrapolation, using the condition that the direction of $\mathbf{n}$ at the boundary should be in accordance with the prescribed contact angle. For instance, if the first row of cells inside the computational region is indexed by $j = 2$, we use the condition that the vector

$$\left(\frac{C(i+1,2) - C(i-1,2)}{2\Delta x}, \frac{C(i,2) - C(i,1)}{\Delta y}\right) \tag{14}$$

should be in the direction of $\mathbf{n}$ to determine $C(i, 1)$. This then allows us to define ∇C at the boundary. If smoothing is used, we apply it iteratively; that is, we use a procedure replacing C by an average involving only neighboring cells and apply this smoothing step several times if we desire. Once $C(i, 1)$ is defined by (14), we can calculate the "once smoothed" function for $j = 2$. Then we repeat the use of (14) to extrapolate the smoothed function to $j = 1$.

The second method mimics the classical argument of Young. We treat the flow as a three-phase system where each phase has its own surface tension force and surface tension coefficient. Fluid 1 has volume fraction C inside the flow domain and zero outside the domain, fluid 2 has volume fraction $1 - C$ inside the flow domain and zero outside, and the solid has volume fraction 0 inside the flow domian and 1 outside. Hence, each of the three phases has a volume fraction defined everywhere. To each of these volume fraction functions, we associate a continuous surface stress and a surface tension coefficient (σ_1 for fluid 1, σ_2 for fluid 2, and σ_3 for the solid). The surface tension between the two fluids is then given by $\sigma_1 + \sigma_2$ and the contact angle is determined by Young's equation

$$(\sigma_1 + \sigma_2)\cos\theta = \sigma_1 - \sigma_2. \tag{15}$$

2.5. *Solution of the Momentum Equation*

The solution of the momentum equation basically follows the method used in prior work. The simultaneous solution of the continuity and momentum equations is computationally expensive. An efficient approximation is provided by a projection method [2], which proceeds as follows. Given the physical quantities at time level n, the first step is to calculate an intermediate velocity $\mathbf{u}^*$ which satisfies

$$\frac{\mathbf{u}^* - \mathbf{u}^n}{\Delta t} = -\mathbf{u}^n \cdot \nabla\mathbf{u}^n + \frac{1}{\rho}(\nabla \cdot (\mu\mathbf{S}) + \mathbf{F})^n. \tag{16}$$

The expression $\mathbf{u}^*$ is not, in general, divergence-free. We then correct $\mathbf{u}^*$ by the pressure term,

$$\frac{\mathbf{u}^{n+1} - \mathbf{u}^*}{\Delta t} = -\frac{\nabla p}{\rho}, \tag{17}$$

where $\mathbf{u}^{n+1}$ at time level $n+1$ satisfies

$$\nabla \cdot \mathbf{u}^{n+1} = 0. \tag{18}$$

The key idea of this projection method is to substitute Eq. (17) into Eq. (18), to obtain a Poisson equation for the pressure

$$\nabla \cdot \left(\frac{\nabla p}{\rho} \right) = \frac{\nabla \cdot \mathbf{u}^*}{\Delta t}. \tag{19}$$

The solution of this equation is the most time-consuming part of the Navier–Stokes solver, and an efficient solution is crucial for the performance of the whole method. Our code SURFER++ uses the multigrid method. The basic idea of the multigrid method is to combine two complementary procedures: one basic iterative method to reduce the high frequency error, and one coarse grid correction step to eliminate the low frequency error. We choose a two-color Gauss–Seidel iterative method because it breaks the dependence between the variables and therefore allows for parallelization of the scheme. We use a Galerkin method to provide a good coarse grid correction.

For determining the intermediate velocity $\mathbf{u}^*$, we use a semi-implicit method as follows:

$$\begin{aligned} \frac{u^* - u^n}{\Delta t} &= \frac{1}{\rho^n}\frac{\partial}{\partial x}(2\mu^n u^*_x) + \frac{1}{\rho^n}\frac{\partial}{\partial y}\left(\mu^n u^*_y + \mu^n v^n_x\right) - u^n u^n_x - v^n u^n_y + \frac{1}{\rho_n}F^n, \\ \frac{v^* - v^n}{\Delta t} &= \frac{1}{\rho^n}\frac{\partial}{\partial x}\left(\mu^n u^n_y + \mu^n v^*_x\right) + \frac{1}{\rho^n}\frac{\partial}{\partial y}(2\mu^n v^*_y) - u^n v^n_x - v^n v^n_y + \frac{1}{\rho_n}G^n. \end{aligned} \tag{20}$$

Here $\mathbf{u} = (u, v)$ and $\mathbf{F} = (F, G)$.

The implementation of boundary conditions at walls had to be changed from the earlier version of our code. If the bottom boundary of the computational domain is between cells $j = 1$ and $j = 2$, then the no slip boundary condition is discretized as $u(i, 1) = -u(i, 2)$. In our original code, this was enforced in an explicit step; effectively we set $u^*(i, 1) = -u^n(i, 2)$. We found that the error associated with the "time delay" thus introduced into the boundary condition led to inaccuracies at the moving contact line which gave invalid results unless very small time steps were chosen. For this reason, we have changed the implementation of the no-slip condition to an implicit one: $u^*(i, 1) = -u^*(i, 2)$.

3. RESULTS AND DISCUSSION

3.1. *Mass Conservation*

The flow domain is the square $[0, 1] \times [0, 1]$, with periodic boundary conditions in the x-direction and no slip boundary conditions at $y = 0$ and $y = 1$. Figure 2(a) shows the

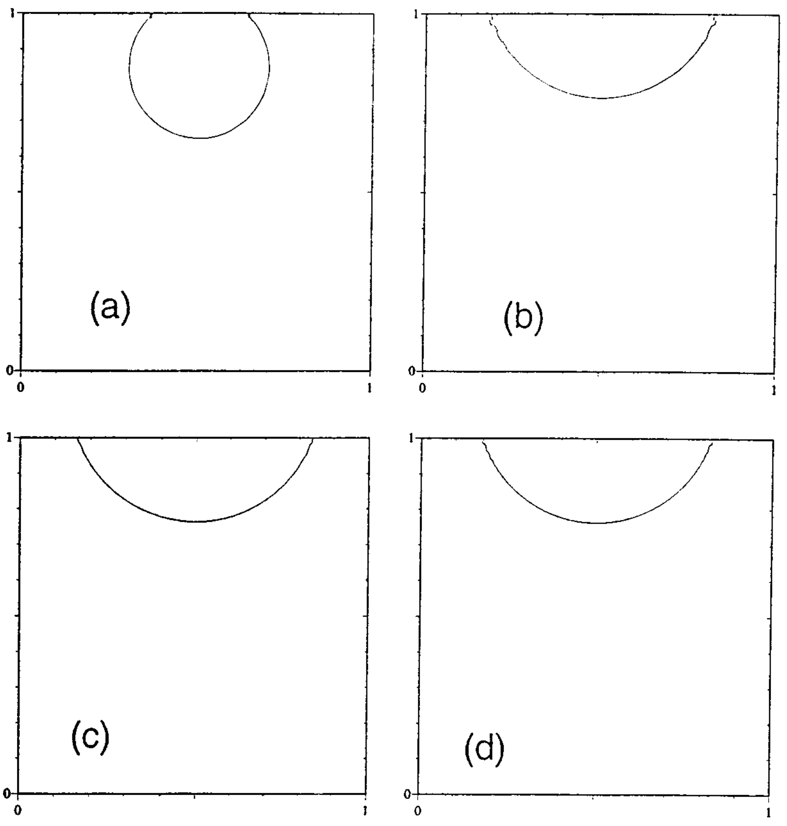

FIG. 2. Computed drop shapes: (a) Initial configuration, (b) $t = 1$, original PLIC scheme, 64×64, (c) $t = 1$, mass-conserving PLIC scheme, 64×64, (d) $t = 1$, original PLIC scheme, 128×128.

initial configuration of the drop. In the initial configuration, fluid 1 occupies the circle of radius $a = 0.2$ centered at (0.5, 0.85) and the fluid is at rest. The viscosity is $\mu = 0.001$ in both fluids, the density of both fluids is $\rho = 0.01$, the interfacial tension is $\sigma = 0.03$, and the contact angle (i.e., the angle between the wall and the interface within fluid 1) is $\arccos(1/3) = 70.53^\circ$. The parameters are not meant to correspond to any specific liquid. What is important are dimensionless quantities. We can form a capillary number,

$$Ca = \frac{\mu U}{\sigma}, \tag{21}$$

and a Reynolds number

$$R = \frac{\rho U a}{\mu}, \tag{22}$$

where U is a velocity scale of the flow. From our computations reported below, we find that the maximum velocity is quite close to 1, leading to a capillary number of about 0.03 and a Reynolds number of about 2.

The initial configuration is in equilibrium except for the contact angle. We compute the spreading of the drop up to time $t = 1$, using a time step of $5 * 10^{-4}$ (the configuration attained at $t = 1$ is actually close to the final configuration). The surface tension stress is

TABLE I
Volumes of Fluid 1 at t = 0, t = 1 for Three Mesh Sizes

Grid	Original volume	Volume at $t = 1$	Percent change
64×64	477.583	450.732	−5.62
128×128	1910.333	1855.439	−2.87
256×256	7641.332	7532.668	−1.42

calculated using the extrapolation method. Table I shows the initial volume of fluid 1 and the volume at $t = 1$ (measured in units of grid squares) for various mesh sizes when the original PLIC scheme is used for interface advection.

In contrast, the mass-conserving PLIC+ scheme described in the previous section yields a final volume of 477.586 for the 64×64 mesh. Clearly, the size of mass conservation errors for the original PLIC scheme is unacceptable for realistic mesh sizes, although the error does decrease linearly with the mesh size.

Figure 2(b) shows the configuration at $t = 1$ computed on the 64×64 mesh with the original PLIC scheme, and Figure 2(c) shows the configuration at $t = 1$ computed on the same mesh with the mass-conserving PLIC+ scheme. Apart from conserving mass, we find a much smoother interface near the contact point (see the enlargement of Fig. 3). Otherwise, the two figures are quite similar. For comparison, Fig. 2(d) shows the final interface computed with the original PLIC scheme on a refined 128×128 grid.

3.2. *Continuous Surface Force Algorithm*

Figures 4 and 5 show a comparison of calculated interface shapes when the extrapolation method is used and when the three-phase method is used. The problem setting is described in the preceding subsection and we show interface positions for $t = 0, 0.05, 0.1, 0.15, 0.2$, and 2. The results from the two methods agree quite well with each other. The position of the contact point on the wall for $t = 2$ is at $x = 0.844$ with the extrapolation method and $x = 0.849$ with the three-phase method. This compares well with the expected value of 0.836 which would be obtained for a drop with the given volume and contact angle (the final equilibrium has been reached at $t = 2$).

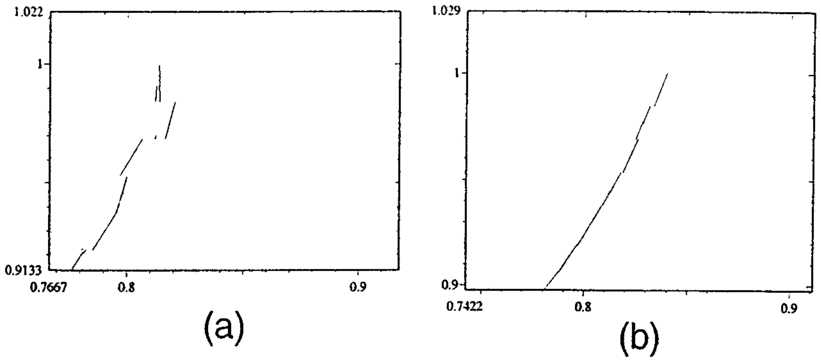

FIG. 3. Interface near contact point at $t = 1$, 64×64 mesh: (a) original PLIC scheme, (b) mass-conserving PLIC+ scheme.

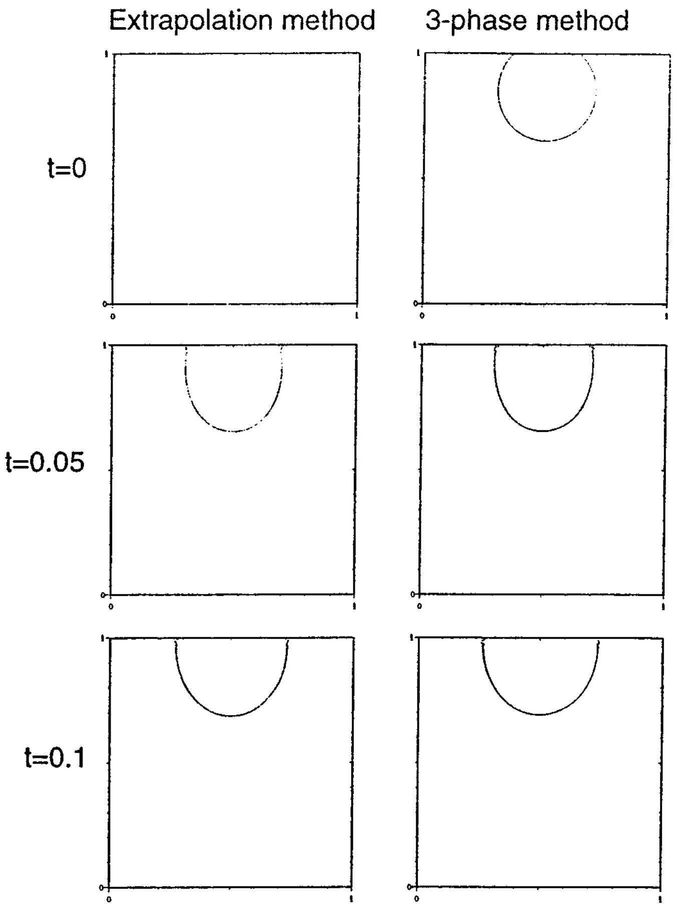

FIG. 4. Comparison of extrapolation and three-phase method (1).

Figure 6 compares velocity fields for $t = 0.1$ and $t = 2$. The velocity field for $t = 0.1$ consists essentially of two counter-rotating vortices which move fluid outward along the top wall and upward along the centerline. The maximum magnitude of the velocity

$$\max(\max |u|, \max |v|) \tag{23}$$

is 0.997 for the extrapolation method and 1.013 for the three-phase method. The velocity fields for $t = 2$ differ substantially. With the extrapolation method, the velocity is very small, with a maximum of 0.025. With the three-phase method, we get a strong local flow near the contact points. The maximum of the velocity is 0.294. We have magnified the flow near the contact point in Fig. 7. The flow is toward the contact point on the walls and the interface and away from the contact point in the interior of each fluid. This flow is driven by the continuous surface force; it attempts to relieve the curvature singularity at the contact point. The difference is that with the extrapolation method, the interface is continued outside the flow region so that there is no curvature singularity. With the three-phase method, on the other hand, the interface is continued along the wall and there is a discontinuity in the slope. There is a basic inconsistency between using a continuous surface force and a sharp interface which is inherent in our numerical method. The persistence of a flow near the contact point

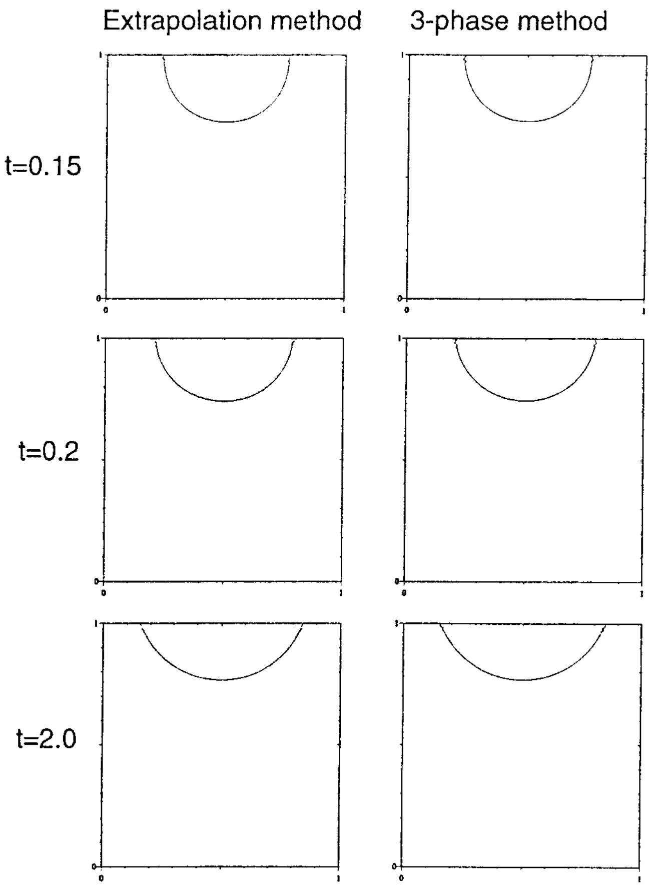

FIG. 5. Comparison of extrapolation and three-phase method (2).

is a result of this inconsistency: the continuous surface force attempts to diffuse the interface to relieve the curvature singularity, but the interface tracking algorithm does not allow the interface to diffuse. Consequently, a static equilibrium is not reached. Our basic conclusion is therefore that the extrapolation method should be preferred.

3.3. *Role of Slip*

Since moving contact lines cause a force singularity, it is generally believed that slip occurs in the neighborhood of the contact line. The simplest law for slip is the Navier slip condition which has the form

$$u = \beta \frac{\partial u}{\partial y}. \tag{24}$$

The slip coefficient β has the dimension of length, which can be interpreted as the distance from the boundary where a linearly extrapolated velocity profile would reach 0. In most applications, this slip length is much smaller than any realistic mesh size for numerical simulation. Numerically, we can impose a slip length on the order of the mesh size; we chose $\beta = \Delta y/2$, since this is particularly easy to implement. Figures 8 and 9 show a comparison of results with slip versus no slip. These results are on a 128×128 mesh with

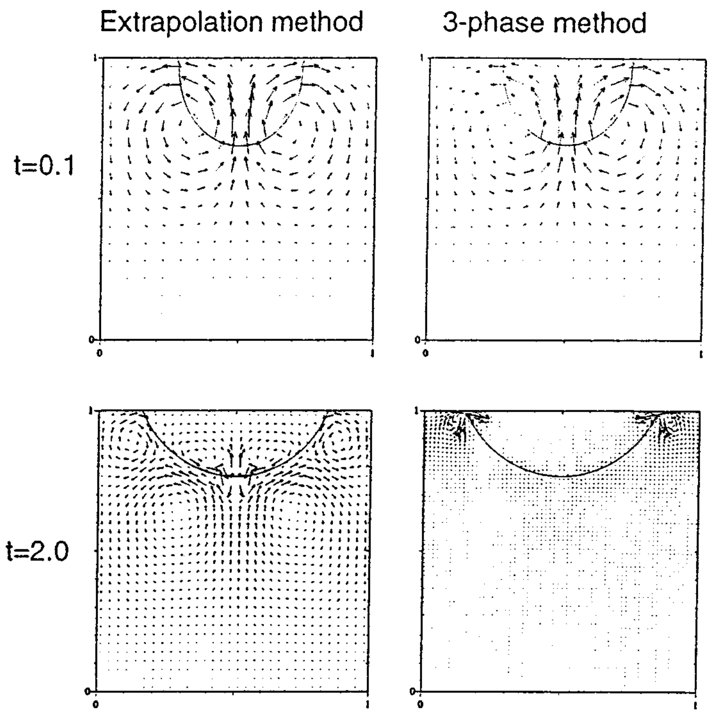

FIG. 6. Velocity fields with extrapolation and three-phase method.

a time step of 10^{-4}. Slip leads to a decrease in the viscous force which resists the motion of the contact line; consequently, the motion of the contact line speeds up.

Table II shows the position of the contact line as a function of time. We also calculated the position at $t = 0.05$ with a 512×512 mesh, the result is 0.672 for no slip and 0.688 for slip. Clearly, mesh refinement leads to a more slowly moving contact line, in both cases.

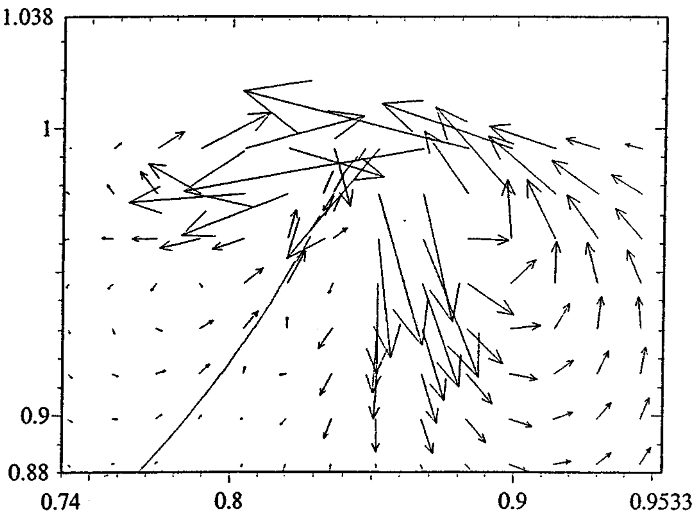

FIG. 7. Local velocity near contact line, $t = 2$.

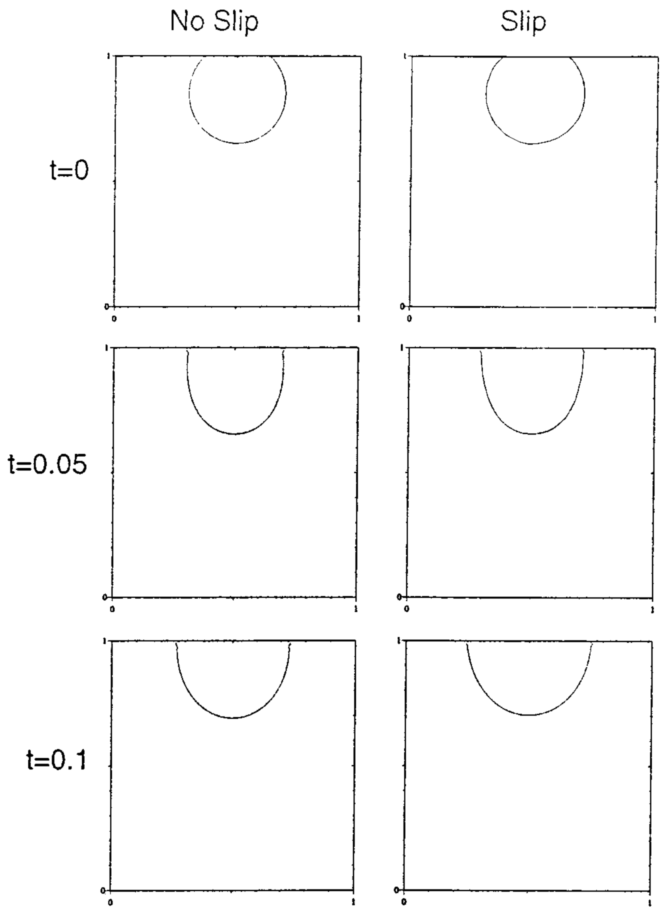

FIG. 8. Interface position with slip and no-slip (1).

This behavior is to be expected. Note that, without slip, it takes an infinite force to move the contact line, while the capillary force driving the flow is finite. If no-slip were strictly enforced, the contact line would be unable to move. Even in the "no-slip" case, our algorithm actually introduces slip on the scale of the mesh. The analysis of Hocking and Rivers [8] predicts a contact line speed which behaves like $a/(b - \ln\lambda)$, where λ is a slip length and a and b depend on the drop radius and on the difference between the actual contact angle and the static equilibrium contact angle. We do not know how the contact angle for our

TABLE II
Position of Contact Line for 128 × 128 and 256 × 256 Mesh vs Time

t	No-slip, 128 × 128	No-slip, 256 × 256	Slip, 128 × 128	Slip, 256 × 256
0	0.633	0.633	0.633	0.633
0.05	0.688	0.681	0.704	0.698
0.1	0.728	0.718	0.750	0.740
0.15	0.762	0.750	0.788	0.776
0.2	0.789	0.776	0.817	0.804
0.25	0.806	0.794	0.829	0.819

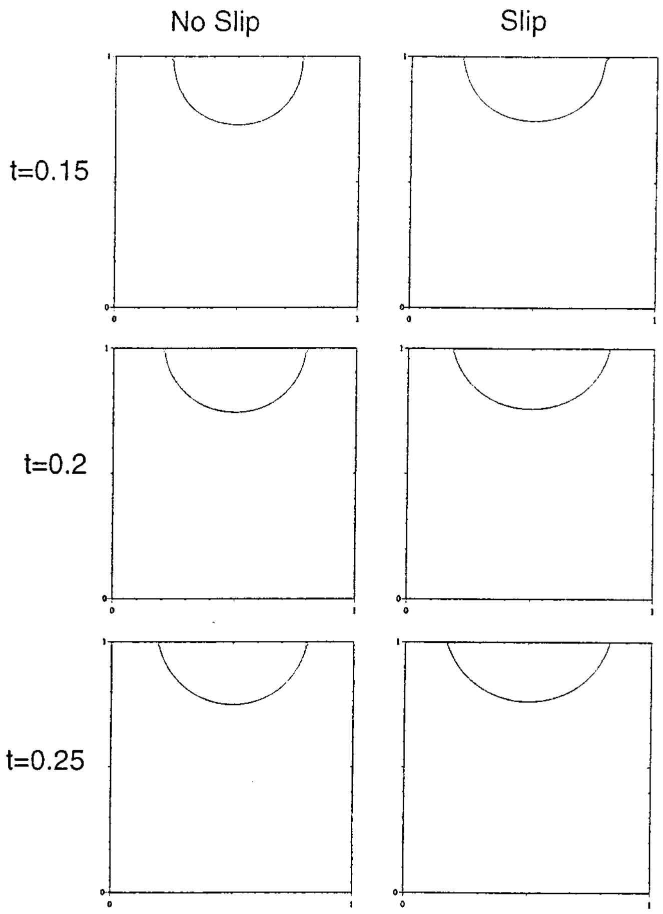

FIG. 9. Interface position with slip and no-slip (2).

problem evolves with time, so we cannot predict a and b as a function of t, but we can expect that, for our numerical simulations, λ is of order h. Our results are consistent with such a prediction. Whether or not we explicitly introduce slip into the boundary condition merely changes the values of a and b.

Since we cannot refine the mesh to a degree in which the true slip length will be captured (our code does not currently include any capability for adaptive mesh refinement), our results will always overpredict the propagation speed of the contact line (because $-\ln \lambda$ is smaller than it should be). This is a fundamental problem independent of the choice of the numerical method. Of course, the problem selected here, where the mismatch of the contact angle is the only effect driving the flow, is a "bad" one in which the size of the slip length is more crucial for the overall flow than in more typical applications. Below, we shall examine a drop stretched by shear, where the primary effect driving the flow is the imposed shear, and contact line motion is a secondary effect. We shall see there that the contact line motion is also mesh dependent, but the overall flow is much less dependent on it.

3.4. *Role of Inertia*

To show the role of inertia, we increase the density by a factor 10 to 0.1. All other parameters are as before (the mesh is 128×128 with a time step of $5 * 10^{-4}$). Figures 10

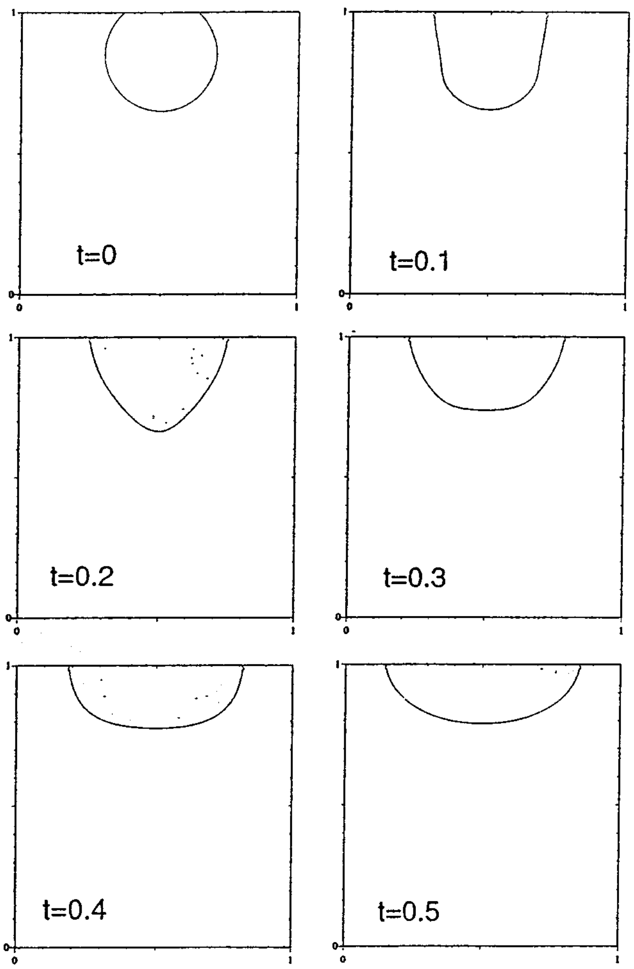

FIG. 10. Interface position with $\rho = 0.1$ (1).

and 11 show the evolution of the interface. Compared to the previous case, we see a slower startup of the motion, followed by an oscillation toward equilibrium. During the transient phase, the drop becomes flatter than before, and the contact line position overshoots the equilibrium position. It then retracts and slightly overshoots again. This exemplifies the role of inertia in adding a resistance to the evolution.

3.5. *Stretching of a Drop Subjected to Shear*

We now impose a shear on the flow by moving the upper boundary at unit speed. The fluid parameters and the initial configuration of the drop are as in Section 3.1 above. The computed interface shapes on a 256×256 mesh and with a time step of $5 * 10^{-4}$ are shown in Fig. 12. The drop is stretched out by the shear in the flow, and the contact line moves by very little (relative to the moving plate). The contact line position depends slightly on mesh refinement, as illustrated in Table III, which shows the position of the right and left contact points as a function of time for two different mesh sizes. In this flow, contact line motion has a minor effect on the overall flow, and the rough mesh is successful in capturing the dynamics.

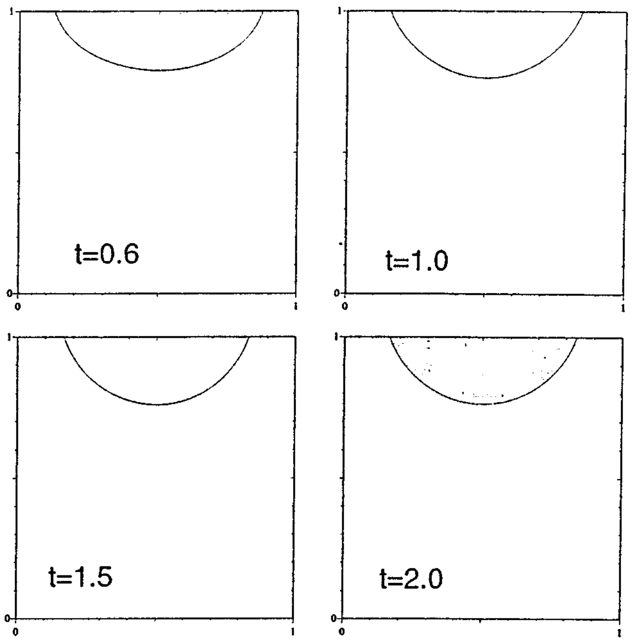

FIG. 11. Interface position with $\rho = 0.1$ (2).

3.6. *Rising Drop*

We now study a drop of one liquid surrounded by another, and rising under gravity. The fluid parameters are as in Section 3.1, except that we now have a density difference between the two liquids. The density in the drop is 0.01, while the density of the surrounding liquid is 0.02. The initial configuration is a circular drop of radius 0.2 centered at a height of 0.75 (as in all the calculations reported here, we are talking about a two-dimensional "drop"). The gravity constant is $g = 50$. This choice is motivated by the balance of gravity and surface tension, which we would like to be of comparable importance. The relevant dimensionless quantity is the Bond number

$$B = \frac{(\delta\rho)ga^2}{\sigma} = 2/3. \tag{25}$$

TABLE III
Position of the Right and Left Contact Points for 256 × 256 and 64 × 64 Mesh

t	right, 256 × 256	left, 256 × 256	right, 64 × 64	left, 64 × 64
0	0.633	0.367	0.633	0.367
0.1	0.735	0.465	0.739	0.459
0.2	0.837	0.563	0.841	0.558
0.3	0.939	0.663	0.942	0.656
0.4	0.040	0.762	0.043	0.754
1.0	0.647	0.361	0.647	0.349
2.0	0.651	0.364	0.649	0.345
3.0	0.651	0.367	0.645	0.343

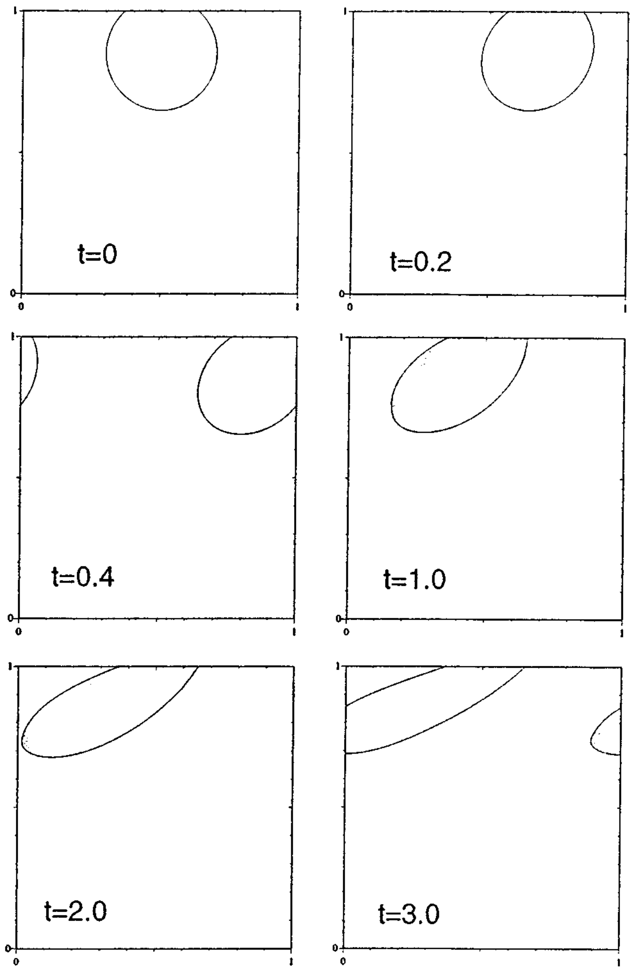

FIG. 12. Drop stretched by shear.

Previous studies of "impact" [5, 6] begin with a spherical drop which is already touching the wall and then spreading. In contrast, our results show that the drop does not reach the top wall in spherical shape, and actually it does not reach it at all. We did calculations on a 128×128, 256×256, and 512×512 mesh. Figure 13 shows interface shapes computed on the 256×256 mesh.

While the drop is rising, it flattens out, and eventually the points closest to the wall are not at the center. Then the drop appears to reach the top at a time between 0.68 and 0.7. This merging depends on the scale of the mesh. With the 128×128 mesh, the time in which the drop reaches the top is between 0.34 and 0.36, and with the 512×512 the drop is still separated from the wall at $t = 1$. Figure 14 shows drop shapes computed with the 512×512 mesh.

The shape of the drop has almost equilibrated between $t = 0.7$ and $t = 1.0$, but the drop slowly gets closer to the top wall. The minimum distance is 0.0100 at $t = 0.7$ and 0.0074 at $t = 1.0$, which is indicative of a proportionality to $1/t$. Eventually, the distance is on the scale of the mesh, and a numerical "interface reconnection" occurs. In reality, such an interface reconnection will occur when the distance reaches scales at which microscopic

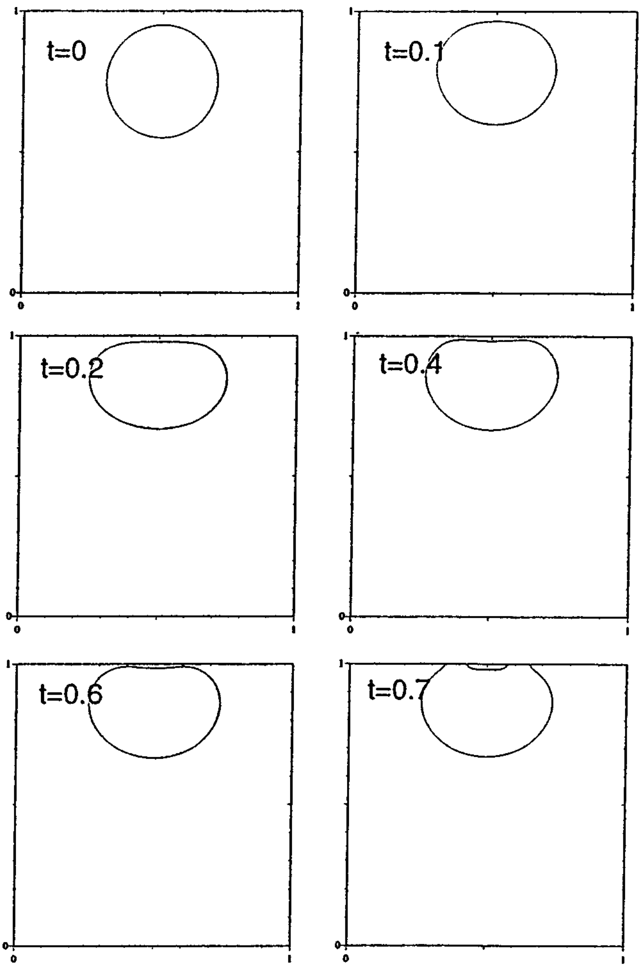

FIG. 13. Rising drop, 256 × 256 mesh.

effects, such as van der Waals forces, become important. Without such forces, the drop would asymptotically approach the wall, but never reach it. The drop is held down against gravity by the pressure that is created by the lubricating flow of liquid being squeezed out above the drop. Since interface reconnection occurs over a small part of the flow domain and a small time period, the precise way in which it occurs is unlikely to have an impact

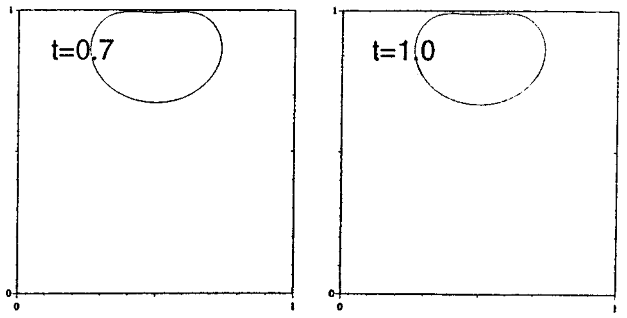

FIG. 14. Rising drop, 512 × 512 mesh.

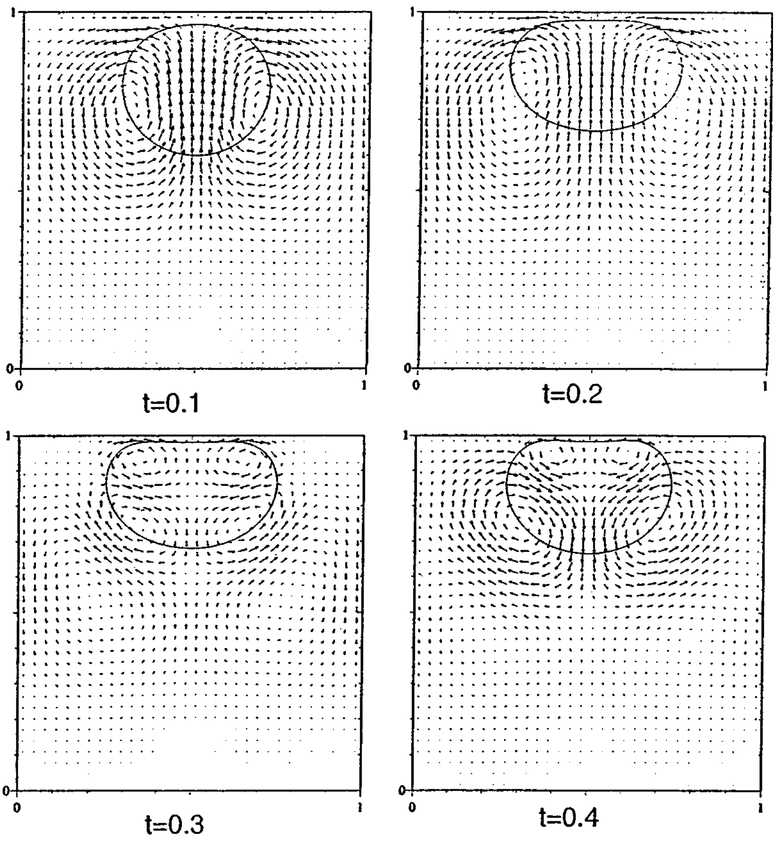

FIG. 15. Velocity field for rising drop.

on the overall flow, but the time at which interface reconnection occurs is crucial, since the flow following interface reconnection is quite different from the flow which precedes it.

Figure 15 shows the development of the velocity field. Initially, the flow consists simply of two counter-rotating vortices. As time evolves, these vortices divide and a more complex flow structure develops.

4. CONCLUSION

We have implemented a moving contact line formulation for the VOF-CSS scheme by suitable reformulations of the PLIC and CSS components. We provide two simple two-dimensional benchmark problems. The first is the transient motion of a drop given an initial contact angle and evolving to an equilibrium state with a new contact angle. The second is a drop falling to the ground under the action of gravity and impacting on it. The performance of the code is encouraging, and the straightforward generalization to the three-dimensional case is a promising novel addition to the VOF method.

ACKNOWLEDGMENT

This research was sponsored by the National Science Foundation under Grants CTS-9612308, NSF-INT 9815106, and DMS-9870220. This work was supported by National Computational Science Alliance under Grants

CTS990010N, CTS990059N, and CTS990063N, and utilized the NCSA SGI Origin 2000. We are grateful to ICAM for the use of their Origin 2000. We thank Osman Basaran for discussions.

REFERENCES

1. J. U. Brackbill, D. B. Kothe, and C. Zemach, A continuum method for modeling surface tension, *J. Comput. Phys.* **100**, 335 (1992).
2. A. J. Chorin, A numerical method for solving incompressible viscous flow problems, *J. Comput. Phys.* **2**, 12 (1967).
3. R. G. Cox, The dynamics of the spreading of liquids on a solid surface. Part 1. Viscous flow, *J. Fluid Mech.* **168**, 169 (1986).
4. E. B. Dussan V, On the spreading of liquids on solid surfaces: static and dynamic contact lines, *Ann. Rev. Fluid Mech.* **11**, 371 (1979).
5. J. Fukai *et al.*, Modeling of the deformation of a liquid droplet impinging upon a flat surface, *Phys. Fluids A* **5**, 2588 (1993).
6. J. Fukai *et al.*, Wetting effects on the spreading of a liquid droplet colliding with a flat surface: Experiment and modeling, *Phys. Fluids* **7**, 236 (1995).
7. P. J. Haley and M. J. Miksis, The effect of the contact line on droplet spreading, *J. Fluid Mech.* **223**, 57 (1991).
8. L. M. Hocking and A. D. Rivers, The spreading of a drop by capillary action, *J. Fluid Mech.* **121**, 425 (1982).
9. D. D. Joseph and Y. Y. Renardy, *Fundamentals of Two-Fluid Dynamics*, 2 vols. (Springer-Verlag, New York, 1993).
10. D. B. Kothe, R. C. Mjolsness, and M. D. Torrey, RIPPLE: A Computer Program for Incompressible Flows with Free Surfaces, Los Alamos National Laboratory Report LA-12007-MS (1991).
11. J. Li, Calcul d'interface affine par morceaux, *C. R. Acad. Sci. Paris* **320**, 391 (1995).
12. J. Li, Y. Renardy, and M. Renardy, A numerical study of periodic disturbances on two-layer Couette flow, *Phys. Fluids* **10**, 3056 (1998).
13. J. Li and Y. Renardy, Direct simulation of unsteady axisymmetric core-annular flow with high viscosity ratio, *J. Fluid Mech.* **391**, 123 (1999).
14. J. Li, Y. Renardy, and M. Renardy, Numerical simulation of breakup of a viscous drop in simple shear flow with a volume-of-fluid method, *Phys. Fluids* **12**, 269 (2000).
15. C. G. Ngan and E. B. Dussan V, On the dynamics of liquid spreading on solid surfaces, *J. Fluid Mech.* **209**, 191 (1989).
16. C. Pozrikidis, The deformation of a liquid drop moving normal to a plane wall, *J. Fluid Mech.* **215**, 331 (1990).
17. R. Scardovelli and S. Zaleski, Direct numerical simulation of free surface and interfacial flow, *Ann. Rev. Fluid Mech.* **31**, 567 (1999).
18. W. J. Silliman and L. E. Scriven, Separating flow near a static contact line: slip at a wall and shape of a free surface, *J. Comput. Phys.* **34**, 287 (1980).