Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Computational Physics 205 (2005) 391–400
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

Short Note

A multi-phase flow method with a fast, geometry-based fluid indicator

Hector D. Ceniceros [a,*], Alexandre M. Roma [b]

[a] *Department of Mathematics, University of California Santa Barbara, CA 93106, USA*
[b] *Department of Applied Mathematics, University of Sao Paulo, Caixa Postal 66281, CEP 05311-970, Sao Paulo-SP, Brazil*

Received 13 April 2004; received in revised form 2 November 2004; accepted 19 November 2004
Available online 19 December 2004

Abstract

We present a novel methodology for incompressible multi-phase flow simulations in which the fluid indicator is a local signed distance (level set) function, and front-tracking is used to evaluate accurately geometric interfacial quantities and forces. Employing ideas from *Computational Geometry*, we propose a procedure in which the level set function is obtained at optimal computational cost without having to solve the level set equation and its associated re-initialization. This new approach is robust and yields an accurate and sharp definition of the distinct bulk phases at all times, irrespective of the geometric complexity of the interfaces. We illustrate the proposed methodology with an example of surface tension-mediated Kelvin–Helmholtz instability.

Keywords: Level set method; Immersed boundary method; Re-initialization; Two-phase flow; Closest point transform

1. Introduction

One important problem in multi-phase flow methods is how to update the fluid material properties (viscosity and density) which might have large discontinuity jumps across the interfaces between different fluids. Since the material properties are constant in each of the bulk phases and the interface motion is limited by the CFL condition to less than a mesh size in each time step, it is computationally appealing to update these quantities only in a vicinity of the interface.

* Corresponding author. Tel.: +805 893 3462; fax: +805 893 2385.
E-mail addresses: hdc@math.ucsb.edu (H.D. Ceniceros), roma@ime.usp.br (A.M. Roma).
URLs: www.math.ucsb.edu/~hdc (H.D. Ceniceros), www.ime.usp.br/~roma (A.M. Roma).

0021-9991/$ - see front matter
doi:10.1016/j.jcp.2004.11.013

Several approaches have been proposed in the literature to address this problem. In front-tracking methods, the simplest procedure would be to sweep the discrete interface element-wise and identify on which side of each element (line segment) the Eulerian grid points next to it appear. However, this straightforward "local" procedure, employed for example by Udaykumar et al. [1], yields incorrect results when two interfaces or two disparate segments of the same interface lie too close to each other [2,3]. To prevent this problem, a more "global" approach, where the whole interface is examined for each Eulerian grid point, must be used. Unverdi and Tryggvason [2] propose a fluid indicator of this type which is cleverly constructed as the solution of a Poisson equation. This equation incorporates the global properties of the interface and can be fast and efficiently solved for typical rectangular domains. However, it must be solved on the entire computational domain and does not take advantage of the fact that the material quantities only change in a vicinity of the interface. Moreover, as reported by Tryggvason et al. [3], this procedure produces oscillations near the fluid interface and inaccuracies away from it.

In the level set method approach [4], the interface is implicitly given ("captured") by the zero level set of a function initialized as the signed distance to the fluid interface. This continuous level set function also serves as a natural fluid indicator for multi-phase incompressible flows and can be updated easily via a simple advection equation. Typically, this equation is solved on the entire domain, although there have been important advances in the design of local level set methods [5,6]. However, it is well known that the level set function can be quickly distorted by the flow and a re-initialization (re-distancing) procedure is needed at every time-step to keep it as a signed distance function around the interface [7]. Another front-capturing approach is the volume-of-fluid method (VOF), in which the interface is reconstructed from the dynamically advected volume fraction, a fluid indicator itself. Some recent implementations yield good accuracy and conservation of mass properties [8]. VOF has also been combined with interface tracking giving rise to "hybrid" methods to obtain this fluid indicator in a vicinity of the interface as proposed by Popinet and Zaleski [9], and Aulisa et al. [10].

Here, we start from yet a different hybrid setting [11] in which the level set function serves to update the fluid material properties and front-tracking (immersed boundary method [12]) couples the fluid interface to the Eulerian domain. We replace the standard procedures for updating the level set function with a fast algorithm from *Computational Geometry* [13]. The resulting new methodology is robust, takes advantage of the local nature of the problem, and is also computationally optimal (linear in the number of Lagrangian markers).

2. Mathematical formulation

To present the new approach, let us consider a single interface separating two incompressible fluids of constant but possibly different density and viscosity and in the presence of (possibly non-uniform) surface tension. The dynamics of this system can be modeled by:

$$\rho(\phi)(\mathbf{u}_t + \mathbf{u}\cdot\nabla\mathbf{u}) = -\nabla p + \nabla\cdot\mu(\phi)(\nabla\mathbf{u} + \nabla\mathbf{u}^{\mathrm{T}}) + \rho(\phi)\mathbf{g} + \mathbf{f}, \tag{1}$$

$$\mathbf{f}(\mathbf{x},\mathbf{t}) = \int \frac{\partial}{\partial\alpha}(T\hat{\mathbf{t}})\delta(\mathbf{x} - \mathbf{X}(\alpha,t))\,\mathrm{d}\alpha, \tag{2}$$

$$\nabla\cdot\mathbf{u} = 0, \tag{3}$$

$$\mathbf{X}_t(\alpha,t) = \int_\Omega \mathbf{u}(\mathbf{x})\delta(\mathbf{x} - \mathbf{X}(\alpha,t))\,\mathrm{d}\mathbf{x}, \tag{4}$$

where ϕ is a fluid indicator (e.g., $\phi > 0$ for one of the fluids and $\phi < 0$ for the other). Here, $\mathbf{u}$, p, and $\mathbf{g}$ are the velocity field, the pressure, and the gravity acceleration, respectively. T is the surface tension coefficient and $\hat{\mathbf{t}}$ is a unit tangent vector to the interface. Note that the interfacial or immersed boundary force $\mathbf{f}$ is obtained from the explicit Lagrangian representation of the immersed boundary $\mathbf{X}(\alpha,t)$, where α is a Lagrangian parameter. The use of the separately tracked interface makes possible a more accurate computation of interfacial quantities, allowing for a better resolution of small scale interfacial phenomena.

Given ϕ, the material quantities are obtained by the relations:

$$\rho(\phi) = \rho_1 + (\rho_2 - \rho_1)H(\phi), \tag{5}$$

$$\mu(\phi) = \mu_1 + (\mu_2 - \mu_1)H(\phi), \tag{6}$$

where ρ_1, ρ_2 and μ_1, μ_2 are the constant densities and viscosities, respectively, and $H(\phi)$ is the Heaviside function defined by

$$H(\phi) = \begin{cases} 0 & \text{if } \phi < 0, \\ 1 & \text{if } \phi \geqslant 0. \end{cases} \tag{7}$$

3. Fast computation of the fluid indicator

Let the interface Γ be represented by a non-self-intersecting, piecewise linear curve. We define ϕ as the signed distance function only in T_γ, a narrow band centered at Γ and of width $2\gamma > 0$. Outside this band, ϕ is continuously defined to be $\pm\gamma$, that is

$$\phi(\mathbf{x}) = \begin{cases} -\gamma & \text{if } d(\mathbf{x}) < -\gamma, \\ d(\mathbf{x}) & \text{if } |\, d(\mathbf{x}) \,| \leqslant +\gamma, \\ +\gamma & \text{if } d(\mathbf{x}) > +\gamma, \end{cases} \tag{8}$$

where the signed distance function $d(\mathbf{x})$ is the Euclidean distance from the given point $\mathbf{x}$ to the fluid interface Γ for which a sign is chosen according to the direction of the normal.

To keep ϕ as a local signed distance function around Γ at all times, we employ a fast algorithm to compute the *Closest Point Transform* (CPT) due to Mauch [13]. The CPT finds the closest point on Γ and determines the Euclidean distance to Γ for all the Eulerian grid points within a specified distance $\epsilon > 0$ from Γ.

Since Γ is a piecewise linear curve, the closest point $\boldsymbol{\xi}$ on Γ to a given point $\mathbf{x}$, either lies on one of the "edges" (links) or at one of the "vertices" (immersed boundary points). If $\boldsymbol{\xi}$ lies on a edge, the vector from $\boldsymbol{\xi}$ to $\mathbf{x}$ is orthogonal to the edge. Thus, the set of closest points to a given edge must lie within a strip defined by the edge itself and by its normal vector. In particular, the set of closest points within a specified distance

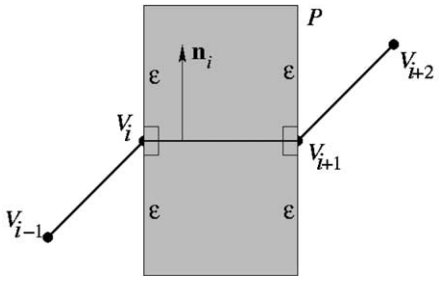

Fig. 1. Polygon P: set of points within distance ϵ to Γ for which the closest point on Γ lies on the edge $V_i \rightarrow V_{i+1}$.

$\epsilon > 0$ to a given edge is given by the polygon P seen in Fig. 1. When $\boldsymbol{\xi}$ lies on a vertex, the vector from $\boldsymbol{\xi}$ to $\mathbf{x}$ must lie between the normal vectors to the two adjacent edges at the vertex. Thus, the closest points to a vertex must lie in a wedge. If the outside (inside) angle formed by the two adjacent edges is less then π then there are no points of positive (negative) distance from the vertex. The set of closest points, within a specified distance $\epsilon > 0$, is *contained* by polygons P like those shown in Fig. 2. The vertex opposite to V_i in Fig. 2(a) is determined by taking the intersection between the lines which are perpendicular to the edges intersecting at V_i (similarly, at V_{i+1} in Fig. 2(b)). Note that, given ϵ such that $0 < \gamma < \epsilon$, the union of all polygons constructed as above will contain the band T_γ.

From the previous considerations, a simple algorithm for updating ϕ in a neighborhood of T_γ at every time step can be devised: Simply put, after updating the location of Γ, we flag all Eulerian grid points $\mathbf{x}_{ij}$ in the union of all polygons (by setting $d(\mathbf{x}_{ij}) = +\infty$) in a first pass. Then, in a second pass, we compute the CPT for all such points $\mathbf{x}_{ij}$, and simultaneously apply the cutoff given by (8). If we let E and V be, respectively, the sets of all edges and of all vertices composing Γ then, for each element q in the union $E \cup V$, this algorithm can be written in pseudo-code as

Algorithm 1. Fast computation of the fluid indicator algorithm

```
for step = 1 to 2 do
  for each q ∈ E ∪ V do
    P ← polygon containing closest points within distance ε to q
    G ← Eulerian grid points inside polygon P
    for each point x_ij∈G do
     if step = 1 then d_ij ← + ∞ end if
     if step = 2 then
              d ← signed distance from x_ij to q
              if |d | < |d_ij|  then
                  d_ij ← d
                  φ_ij ← sign(d_ij)*min{|d_ij|,γ}
              end if
      end if
    end for
  end for
end for
```

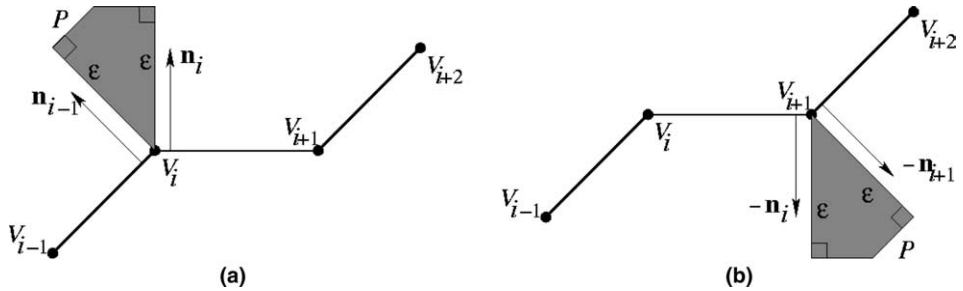

Fig. 2. Polygons containing the set of points within distance ϵ to Γ for which the closest point on Γ lies on either the vertex V_i or V_{i+1} [(a) or (b), respectively].

We emphasize that Algorithm 1 is executed at every time step, after the interface position has been updated. Note also that at $t = 0$, we must have a fluid indicator function ϕ satisfying (8) so that only a local correction is needed subsequently. This initial ϕ can be obtained by computing at $t = 0$ the signed distance at every point in the Eulerian grid and then by applying the cutoff to it.

To determine G, the set of Eulerian grid points inside P, we start by finding a *bounding box* around P, that is, the smallest possible rectangle aligned with the coordinate axes containing P, whose vertices are grid points. We then proceed by checking whether or not each Eulerian point in the bounding box is interior to the polygon by applying the *Point Containment Algorithm* described in Appendix A.

Finally, we observe that Algorithm 1 has linear computational complexity in the number of interfacial markers and, since it takes into account the full extent of the free boundary to compute the signed distance, it correctly handles situations when two disparate interface segments lie too close to each other, such as in the near merging or in the near self-intersection cases.

4. Numerical scheme

We illustrate the proposed methodology by solving (1)–(4) with an example of a surface tension mediated Kelvin–Helmholtz instability problem when the viscosity is constant and the gravity effects are neglected. Assuming that $\Delta x = \Delta y = h$ for simplicity, we employ on a uniform MAC grid a variable-density variant of the predictor–corrector scheme introduced in [14], given by

$$\rho^{n+\frac{1}{2},m-1} = \rho_1 + (\rho_2 - \rho_1)H(\phi^{n+\frac{1}{2},m-1}), \tag{9}$$

$$\frac{\mathbf{u}^{*,m} - \mathbf{u}^n}{\Delta t} + \frac{\mathbf{G}p^{n+\frac{1}{2},m-1}}{\rho^{n+\frac{1}{2},m-1}} = \frac{\mu}{\rho^{n+\frac{1}{2},m-1}} L\left(\frac{\mathbf{u}^{*,m} + \mathbf{u}^n}{2}\right) - [(\mathbf{u}\cdot\nabla)\mathbf{u}]^{n+\frac{1}{2},m-1} + \frac{\mathbf{f}^{n+\frac{1}{2},m-1}}{\rho^{n+\frac{1}{2},m-1}}, \tag{10}$$

$$\frac{\mathbf{u}^{n+1,m} - \mathbf{u}^n}{\Delta t} + \frac{\mathbf{G}p^{n+\frac{1}{2},m}}{\rho^{n+\frac{1}{2},m-1}} = \frac{\mathbf{u}^{*,m} - \mathbf{u}^n}{\Delta t} + \frac{\mathbf{G}p^{n+\frac{1}{2},m-1}}{\rho^{n+\frac{1}{2},m-1}}, \tag{11}$$

$$\mathbf{D}\cdot\mathbf{u}^{n+1,m} = 0, \tag{12}$$

$$\frac{\mathbf{X}^{n+1,m} - \mathbf{X}^n}{\Delta t} = h^2 \sum_{\mathbf{x}} \left(\frac{\mathbf{U}^{n+1,m} + \mathbf{U}^n}{2}\right), \tag{13}$$

$$\phi^{n+1,m} \doteq \phi(\mathbf{X}^{n+1,m}), \tag{14}$$

where $\phi^{n+\frac{1}{2},k} \doteq \frac{1}{2}[\phi(\mathbf{X}^{n+1,k}) + \phi(\mathbf{X}^n)]$ with each term being obtained by application of Algorithm 1, and the velocities by

$$\mathbf{U}^{n+1,m} \doteq \mathbf{u}^{n+1,m}\delta_h(\mathbf{x} - \mathbf{X}^{n+1,m-1}) + U_A^{n+1,m}\hat{\mathbf{t}}^{n+1,m-1}. \tag{15}$$

In (15), U_A is a tangential velocity which is conveniently selected in order to dynamically control the fluid interface mesh spacing, and $\hat{\mathbf{t}}$ is the tangent vector which is given by

$$\hat{\mathbf{t}}^{n+1,m-1} = \frac{D_{\Delta\alpha}\mathbf{X}^{n+1,m-1}}{\|D_{\Delta\alpha}\mathbf{X}^{n+1,m-1}\|}, \tag{16}$$

where $\Delta\alpha$ is the mesh spacing in the parametrizing variable and $D_{\Delta\alpha}$ is the centered difference operator in α.

The interface position is approximated by a piecewise linear representation, and the Dirac δ function is approximated by a mollified version δ_h. There are many possible choices for this function. Here, we choose Peskin's delta [12],

$$\delta_h(\mathbf{x}_{\mathbf{i},\mathbf{j}}) = d_h(x_i)d_h(y_j), \tag{17}$$

where

$$d_h(z) = \begin{cases} 0.25[1+\cos(\frac{\pi}{2}z/h)]/h & \text{for } |z| < 2h, \\ 0 & \text{for } |z| \geqslant 2h. \end{cases} \tag{18}$$

This choice for $\delta_h(\mathbf{x})$ provides good regularization properties around the interface and it is motivated by a set of compatibility properties described by Peskin [12].

The force in (10) is approximated by $\mathbf{f}^{n+\frac{1}{2},m-1} \doteq \frac{1}{2}[\mathbf{f}^{n+1,m-1} + \mathbf{f}^n]$. Since the flow is assumed to take place in a channel whose walls move in opposite directions, this force, besides the surface tension component given by (2), has another component originating at the walls. The contribution coming from the surface tension is computed by $\Delta\alpha\sum_k TD_{\Delta\alpha}\hat{\mathbf{t}}_k\delta_h(\mathbf{x}-\mathbf{X}_k)$. We refer the reader to [14] for detailed discussions on the aspects of the numerical scheme and on the mathematical modeling not covered here.

We have the predictor step when $m = 1$ and the corrector step when $m = 2$. This scheme is formally second-order for smooth solutions but in the presence of singular forces, it would be only first-order accurate near the immersed boundaries. Although the flow solver is an important part in the overall method, note that the proposed methodology of this work can be implemented with any other Navier–Stokes solver.

We employ the diffused interface setting of the immersed boundary method to compute the interfacial forces which are smeared out by employing a delta function approximation. However, the same ideas could also be applied in a sharp interface approach. For example, if the viscosity is continuous, the recent immersed interface method [15] can easily be implemented. It would only require a local modification to the right hand side of the Poisson equation for the pressure and to determine the fluid velocity on the interface by simple interpolation. Discontinuous viscosity is also possible along the lines of, for example, the ghost fluid method [16].

Finally, we note that, by employing the Heaviside function (7), the density is treated as a discontinuous function which, in addition to the singular forces, lowers the accuracy near the immersed boundaries.

5. Numerical example: Kelvin–Helmholtz instability

We apply the proposed methodology to compute the dynamics of a sheared interface between two incompressible fluids which is subjected to surface tension. The domain is a channel with periodic boundary conditions in the stream-wise direction and no-slip boundary conditions on the bounding walls.

The flow can be characterized by the ratios of the constant densities and viscosities

$$\xi = \frac{\rho_2}{\rho_1} = 0.5, \quad \eta = \frac{\mu_2}{\mu_1} = 1.0, \tag{19}$$

where the subscripts 1 (2) corresponds to the fluid below (above) the interface, and by a Reynolds number Re and a Weber number We associated to one of the fluids

$$Re = \frac{\rho_1 \lambda U_c}{\mu_1} = 5000, \quad We = \frac{\rho_1 \lambda U_c^2}{T} = 400, \tag{20}$$

where λ and U_c are characteristic length and velocity, respectively and T is the surface tension. We define the length scale λ as the periodicity length of the channel and the velocity scale U_c as the difference between the horizontal velocities at the walls.

The computational domain is $\Omega_C = [0,1] \times [-1,1]$. The rigid walls are placed at $y = \pm 1.00 \mp 0.125$ and move in opposite directions at constant speeds ∓ 0.500, respectively.

Initially, the fluid interface is given in parametric form by

$$\mathbf{X}_0(\alpha) = (\alpha + 0.01 \sin 2\pi\alpha, -0.01 \sin 2\pi\alpha), \quad 0 \leqslant \alpha \leqslant 1. \tag{21}$$

We obtain the initial velocity (u_0, v_0) from a delta supported vorticity with unit strength

$$\omega_0(x,y) = \delta_h(y + 0.01 \sin 2\pi(x+y)), \quad (x,y) \in \Omega_C, \tag{22}$$

where δ_h is given by (18). With this vorticity, we solve the Poisson equation

$$\Delta\psi = -\omega_0 \tag{23}$$

in Ω_c with periodic boundary conditions in the stream-wise direction and Dirichlet homogeneous conditions in the normal-wall direction. The initial velocity is then given by

$$u_0(\mathbf{x}) = +\frac{\partial\psi}{\partial y}(\mathbf{x}), \quad v_0(\mathbf{x}) = -\frac{\partial\psi}{\partial x}(\mathbf{x}). \tag{24}$$

To demonstrate the numerical convergence of the overall predictor–corrector scheme (9)–(14), we compute the solution up to $t = 3.00$ on three different uniform grids: 128×256, 256×512, and 512×1024. In each case, the time step is chosen respecting the first-order CFL condition. The number of markers employed to define the interface in each run is variable and dynamically controlled in a manner to ensure that we have the distance between two consecutive markers, Δs, satisfying $\Delta x/2 \leqslant \Delta s \leqslant \Delta x$.

We approximate the L_2 errors in two sub-domains, $\Omega_1 = [0,1] \times [-0.35,0.35]$ and $\Omega_2 = [0,1] \times ([-0.65,-0.45] \cup [0.45,0.65])$, by averaging the finer grid solution onto the coarser one. These estimates are used to compute the rates of convergence, given in Table 1.

The results indicate that in a vicinity of the fluid interface (on Ω_1) the method performs near or at first-order accuracy, an inheritance from the immersed boundary method. Away from any immersed boundaries (on Ω_2), where the flow has more regularity, the performance goes up to or near to its formal second-order accuracy. Fig. 3, on the left, shows the fluid interface on a 256×512 and on a 512×1024 resolutions at time $t = 3.00$ (dashed and solid lines, respectively). The small difference between the interface profiles provides

Table 1
L_2 error approximations and convergence rates in the strips Ω_1 and Ω_2

n	128	256	Rate
$\|\mathbf{u}_{n\times 2n} - \mathbf{u}_{2n\times 4n}\|_{2,\Omega_1}$	1.0523×10^{-2}	6.0621×10^{-3}	0.79
$\|\mathbf{u}_{n\times 2n} - \mathbf{u}_{2n\times 4n}\|_{2,\Omega_2}$	1.6971×10^{-4}	5.4873×10^{-5}	1.63

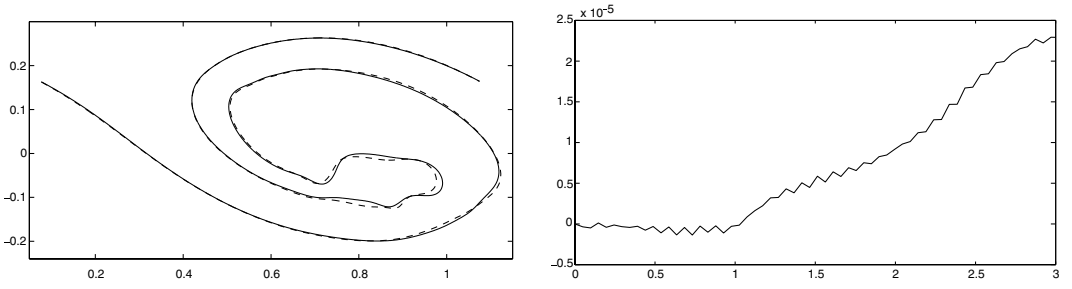

Fig. 3. Left: Interface profiles at $t = 3.00$ for the 256×512 and the 512×1024 resolutions (dashed and solid lines, respectively). Right: Mass variation of the bottom fluid relative to its initial mass (resolution 512×1024).

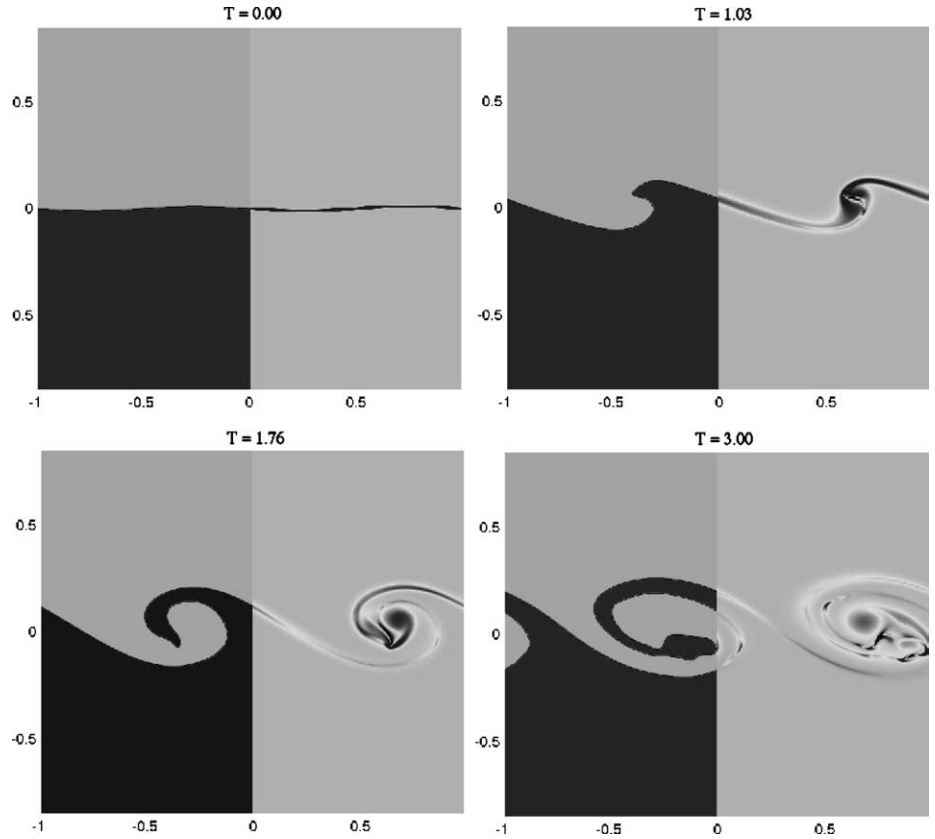

Fig. 4. Fluid indicator and vorticity (on the left and on the right of each plot, respectively) on a 512×1024 grid at different times. $We = 400$, $Re = 5000$, density ratio $\xi = 0.50$, viscosity ratio $\eta = 1.0$.

evidence that the flow on 512×1024 is well resolved. On the right, Fig. 3 shows the relative change of the area covered by the lower fluid, $(A(t) - A_0)/A_0$, where A_0 is the initial area. Note that, for the 512×1024 resolution, it implies that mass is conserved with a maximum relative change less than 2.5×10^{-3} %.

Fig. 4 depicts the flow evolution in time. On the left of each plot, we have the fluid indicator ϕ and, on its rigth, the vorticity, both given as flooded contour plots. Small, asymmetric fluid fingers develop at early times ($t = 1.03$) and these subsequently roll up. The rolled-up finger corresponding to the heavier phase (dark shading) is much thinner than that of the lighter phase. In each time step, the fluid indicator function ϕ is updated only locally in a vicinity T_γ of the interface whose radius $\gamma = 2\sqrt{2}\Delta x$. By construction, ϕ is always continuous and specifies sharply the material properties of the two bulk phases. The vorticity which initially is solely concentrated around the fluid interface subsequently diffuses into the bulk phases. At later times ($t = 1.76$ and $t = 3.00$), large amounts vorticity concentrate near the points of highest interfacial curvature.

6. Concluding remarks

We proposed a method for incompressible multi-phase flow in which the fluid indicator is a local signed distance function (level set) while front-tracking is used to evaluate accurately geometric interfacial quantities and forces. The main contribution of the present work is the application of a novel technique from

Computational Geometry [13] to update efficiently the signed distance to a piecewise linear approximation of the fluid interface. At each time step, the fluid indicator (hence the fluid material properties) is updated only locally, in a thin band surrounding the interface, at an optimal computational cost $O(N_b)$, where N_b is the number of interface markers. The proposed methodology is robust and relatively simple to implement, allowing to work with sharp (discontinuous) material quantities. This approach avoids entirely solving the level set equation and its associated re-initialization.

We emphasize that the ideas presented here can be implemented with any Navier–Stokes solver derived from similar mathematical formulations, and be efficiently extended to 3D, as long as triangular meshes are employed in the discretization of the fluid interface [13]. As a future work, we envisage the possibility of employing other elementary techniques from *Computational Geometry* in the simulation of fluid interfaces (such as determining whether or not a marker chain will self intersect) for dealing with, for example, changes in the interface topology (e.g., merging, pinching-off and reconnection), which are not currently handled by our present implementation.

Acknowledgements

The authors thank Dr. A.E. Fabris and Dr. J.C. de Pina of IME-USP for their helpful insights in the area of Computational Geometry. Partial support for this research was provided by the National Science Foundation under Grant No. DMS-0311911. H.D.C. also acknowledges support under a UCSB Faculty Career Development Award Grant.

Appendix A. The point containment algorithm

Given an oriented line segment from $V_i = (x_i,y_i)$ to $V_{i+1} = (x_{i+1},y_{i+1})$ and a point $P = (x_p,y_p)$, the number

$$c_z = (y_p - y_i)(x_{i+1} - x_i) - (x_p - x_i)(y_{i+1} - y_i) \tag{A.1}$$

is the component in the z-direction of the cross product $(V_{i+1} - V_i) \times (P - V_i)$. If the result is positive, P is to the left of the line segment, if it is negative then P is to its right, and P is on the line segment if the result is zero.

Assuming that the four vertices of our polygons were ordered in the counterclockwise direction, we use the idea above in Algorithm 2.

Algorithm 2. Point containment algorithm (left-on-right strategy)

```
inside ← false
if isleft(P,V1,V2) then
   if isleft(P,V2,V3) then
      if isleft(P,V3,V4) then
       if isleft(P,V4,V1) then
                inside ← true
       end if
      end if
   end if
end if
```

The logical function "isleft(P,Q,R)" computes (A.1), returning *true* if P is to the left side of the oriented line segment from Q to R, or *false* otherwise.

Note that the order in which the polygon edges are tested affects the speed of Algorithm 2 The polygons employed by Algorithm 1 usually have aspect ratio approximately 1:8 (they are thin and long, roughly $0.5\Delta x \times 4\Delta x$). Since they are seldom aligned with the coordinate axes, their bounding boxes are relatively big and the majority of the Eulerian grid points will lie outside the polygons. In this situation, we greatly benefit by looking for an early exit of the test due to the point being outside the polygon. This can be accomplished if the edges are sorted according to their size (by construction). By testing the biggest edges first (they cut off the most area of the bounding box), our numerical experiments indicate that the computational cost of classifying a grid point inside the bounding box (interior/exterior to the polygon) is *less than that of four multiplications* in the average.

References

[1] H.S. Udaykumar, H.-C. Kan, W. Shyy, R. Tran-Son-Tay, Multiphase dynamics in arbitrary geometries on fixed cartesian grids, J. Comput. Phys. 137 (1997) 366–405.
[2] S.O. Unverdi, G. Tryggvason, A front-tracking method for viscous, incompressible, multifluid flows, J. Comput. Phys. 100 (1992) 25–37.
[3] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, Y.-J. Jan, A front-tracking method for computations of multiphase flow, J. Comput. Phys. 169 (2001) 708–759.
[4] S. Osher, J.A. Sethian, Fronts propagating with curvature dependent speed: algorithms based on Hamilton–Jacobi formulations, J. Comput. Phys. 79 (1988) 12–49.
[5] D. Adalsteinsson, J.A. Sethian, A fast level set method for propagating interfaces, J. Comput. Phys. 118 (2) (1995) 269–277.
[6] D. Peng, B. Merriman, S. Osher, H. Zhao, M. Kang, A PDE-based fast local level set method, J. Comput. Phys. 155 (1999) 410–438.
[7] M. Sussman, P. Smereka, S. Osher, A level set approach for computing solutions to incompressible two-phase flow, J. Comput. Phys. 114 (1994) 146–159.
[8] P.G.J. López, J. Hernández, F. Faura, A volume of fluid method based on multidimensional advection and spline interface reconstruction, J. Comput. Phys. 195 (2004) 718–742.
[9] S. Popinet, S. Zaleski, A front-tracking algorithm for accurate representation of surface tension, Int. J. Numer. Meth. Fluids 30 (1999) 775–793.
[10] S.M.E. Aulisa, R. Scardovelli, A mixed markers and volume-of-fluid method for the reconstruction and advection of interfaces in two-phase and free-boundary flows, J. Comput. Phys. 188 (2003) 611–639.
[11] H.D. Ceniceros, The effect of surfactants on the formation and evolution of capillary waves, Phys. Fluids 15 (1) (2003) 245–256.
[12] C.S. Peskin, Numerical analysis of blood flow in the heart, J. Comput. Phys. 25 (1977) 220–252.
[13] S. Mauch, Efficient algorithms for solving static Hamilton–Jacobi equations. Ph.D. thesis, California Institute of Technology, 2003.
[14] H.D. Ceniceros, A.M. Roma, Study of long-time dynamics of a viscous vortex sheet with a fully adaptive non-stiff method, Phys. Fluids 16 (12) (2004) 4285–4318.
[15] L. Lee, R.J. LeVeque, An immersed interface method for incompressible Navier–Stokes equations, SIAM J. Sci. Comput. 25 (3) (2003) 832–856.
[16] M. Kang, R. Fedkiw, X.-D. Liu, A boundary condition capturing method for multiphase incompressible flow, J. Sci. Comput. 15 (2000) 323–360.