ACADEMIC PRESS

Available online at www.sciencedirect.com

SCIENCE DIRECT®

Journal of Computational Physics 187 (2003) 110–136

JOURNAL OF COMPUTATIONAL PHYSICS

www.elsevier.com/locate/jcp

A second order coupled level set and volume-of-fluid method for computing growth and collapse of vapor bubbles

Mark Sussman [*]

Florida State University, Tallahassee, FL 32306, USA

Received 29 January 2002; received in revised form 10 January 2003; accepted 6 February 2003

Abstract

We present a coupled level set/volume-of-fluid method for computing growth and collapse of vapor bubbles. The liquid is assumed incompressible and the vapor is assumed to have constant pressure in space. Second order algorithms are used for finding "mass conserving" extension velocities, for discretizing the local interfacial curvature and also for the discretization of the cell-centered projection step. Convergence studies are given that demonstrate this second order accuracy. Examples are provided that apply to cavitating bubbles.

Keywords: Level set; Volume-of-fluid; Bubbles; Cavitation

1. Introduction

This paper describes a numerical model for growth and collapse of vapor bubbles using a coupled level set and volume-of-fluid (CLSVOF) method. The applications which motivate this work are ship hydrodynamics, where one wants to measure the effects of an underwater explosion near a ship [7,33], and thermal ink-jet devices where a vapor bubble is created which ejects ink [14].

Previous work in this area fall in the class of boundary integral methods [7,36], level set methods [9,16,28], volume-of-fluid methods [11], marker particle methods [10], front tracking methods [20], and front capturing methods [33].

Each of the above referenced methods has advantages and disadvantages. Boundary integral methods [7,36] discretize only the interface separating liquid and vapor, making these methods very efficient. On the other hand, the boundary integral approach assumes that the solution is governed by potential flow in each fluid; therefore, boundary integral methods are limited in how they model viscosity. Also, since the boundary integral method is a Lagrangian approach, every time the interface merges or splits, one must implement a complicated surgery in order to continue the computation [7,31]. The following Eulerian based

[*] Fax: 1-850-644-7194.
E-mail address: sussman@math.fsu.edu.

0021-9991/03/$ - see front matter © 2003 Elsevier Science B.V. All rights reserved.
doi:10.1016/S0021-9991(03)00087-1

methods [9,11,16,20,28,33], discretize the whole computational domain. These methods are applicable to flows with complex fluid interfaces and other general situations. Unfortunately, all of these Eulerian based schemes use a first order treatment at the liquid vapor interface, which necessitates a finer grid; thus degrading efficiency.

Remarks:

- There have been recent developments for second order *two-fluid* level set methods in which both fluids are handled as incompressible [18]. It should be noted though, that the method proposed by Helenbrook et al. [18] was not applied to flows in which the interface developed points or cusps or in which there was interfacial merging or splitting.
- The SUMMAC method [10] uses second order methods for treating the free surface boundary condition for the computation of finite amplitude water waves. Nonetheless, the SUMMAC method is not second order accurate overall since the temporal derivative is approximated using a first order method. Additionally, the SUMMAC method was not designed for complicated flows (e.g., wave breaking) and the SUMMAC method does not include surface tension.

Our method represents an advance in methodology by "robustly" computing free surface flows to second order accuracy. We shall demonstrate overall second order accuracy before and *after* a change in interfacial topology.

We attain overall second order accuracy by making sure that the following components of our discretization are done with second order accuracy:

- Second order Runge–Kutta time discretization for the momentum equations.
- Coupled level set and volume-of-fluid advection; we shall use second order operator split (Strang-splitting) advection algorithms for both the level set function and the volume fractions [30].
- The interfacial curvature at the zero level set shall be computed to second order accuracy. Our discretization for curvature is based on discretization described by Helmsen et al. [19] for the application of photolithography.
- The Dirichlet pressure boundary condition at the free surface shall be enforced to second order accuracy. Our discretization of this boundary condition is based on the discretization described by Gibou et al. [17] for solving the Poisson equation in irregular domains. We remark, that the matrix system that results

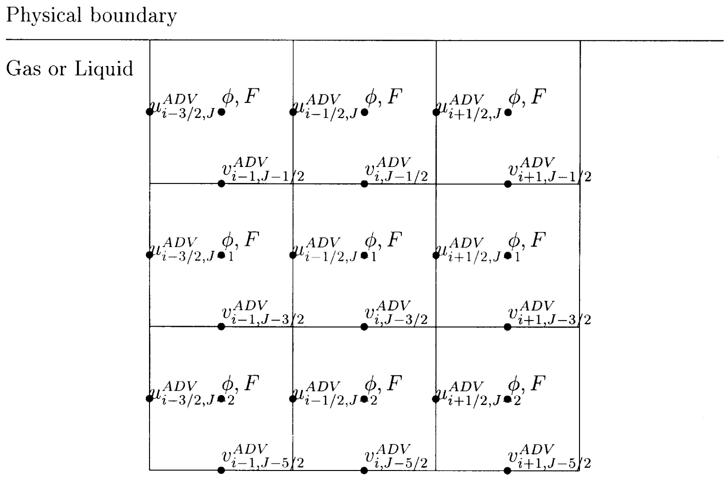
Physical boundary
Gas or Liquid
$u^{ADV}_{i-3/2,J}$ ϕ, F
$u^{ADV}_{i-1/2,J}$ ϕ, F
$u^{ADV}_{i+1/2,J}$ ϕ, F
$v^{ADV}_{i-1,J-1/2}$
$v^{ADV}_{i,J-1/2}$
$v^{ADV}_{i+1,J-1/2}$
$u^{ADV}_{i-3/2,J-1}$ ϕ, F
$u^{ADV}_{i-1/2,J-1}$ ϕ, F
$u^{ADV}_{i+1/2,J-1}$ ϕ, F
$v^{ADV}_{i-1,J-3/2}$
$v^{ADV}_{i,J-3/2}$
$v^{ADV}_{i+1,J-3/2}$
$u^{ADV}_{i-3/2,J-2}$ ϕ, F
$u^{ADV}_{i-1/2,J-2}$ ϕ, F
$u^{ADV}_{i+1/2,J-2}$ ϕ, F
$v^{ADV}_{i-1,J-5/2}$
$v^{ADV}_{i,J-5/2}$
$v^{ADV}_{i+1,J-5/2}$

Fig. 1. Diagram of where the discretely divergence-free face-centered velocity field $\boldsymbol{U}^{\mathrm{ADV}}$, level set function ϕ and volume-of-fluid function F are located in relation to the computational grid and the physical boundary.

from applying the methodology in [17] is symmetric; thus this system can be "robustly" solved using a preconditioned conjugate gradient method for any free surface configuration.
- We shall introduce a new velocity extension procedure for creating second order, mass conserving extension velocities.

Besides introducing a new numerical method for "robustly" treating interfacial flows to second order accuracy, we also introduce a new procedure for carrying out the "approximate" projection step. In our algorithm, the discrete vector field to be projected is located at cell centers, but the projected vector field is located at face centers ("MAC" grid locations). This allows us to accurately implement a "cell-centered" projection step while making it convenient to implement already existing methods for cell-centered implicit viscous solves, second order Cartesian grid methods, parallelization and adaptive mesh refinement. Background information regarding cell-centered projection versus node based projection and cell-centered velocity field versus face-centered velocity field can be found in [2,3,21,22,26].

2. Governing equations

We are given two fluids, liquid and vapor. The liquid is assumed constant density and incompressible, and the vapor is assumed to have zero density and constant pressure in space (but not necessarily in time). Without loss of generality, we shall assume that the liquid density is one. In this paper, we shall also assume that the viscosity is zero in both the liquid and vapor.

In the liquid we have:

$$\boldsymbol{U}_t + \boldsymbol{F}_x + \boldsymbol{G}_y = -\nabla p + \boldsymbol{H}, \tag{1}$$

$$\nabla \cdot \boldsymbol{U} = 0. \tag{2}$$

$$\boldsymbol{F} = \begin{pmatrix} u^2 \\ uv \end{pmatrix}, \quad \boldsymbol{G} = \begin{pmatrix} uv \\ v^2 \end{pmatrix}.$$

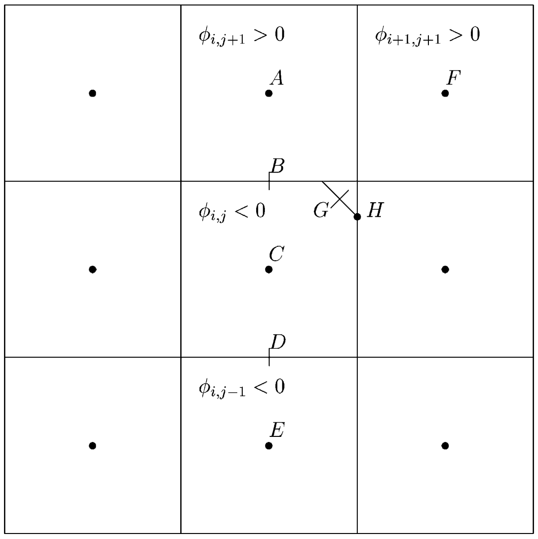

Fig. 2. The distances d assigned to points A, F, and E, respectively, are $|\bar{A}B|$, $|\bar{F}G|$ and $|\bar{E}H|$.

In the vapor, we assume the pressure $p(t)$ is constant in space. For example Szymczak et al. [33] used the following model for an expanding and collapsing adiabatic vapor bubble:

$$p_{\text{vapor}}(t) = p_0 \left(\frac{V(0)}{V(t)} \right)^{\gamma}, \tag{3}$$

where $V(t)$ is the volume of the bubble. The position of the free surface is updated via the level set equation

$$\phi_t + \boldsymbol{U} \cdot \nabla\phi = 0, \tag{4}$$

where ϕ is the level set function which is positive in the liquid and negative in the vapor. The governing equation for the level set function (4) states that ϕ remains constant on particle paths; i.e., if the zero level set is initialized as the free-surface between the liquid and vapor, then the zero level set will always represent the free-surface. From the level set function, one can enforce second order Dirichlet pressure boundary conditions at the free surface. The vapor viscosity μ_{vapor} is assumed to be zero. The free surface boundary conditions are enforced by specifying the following pressure boundary condition at the free surface:

$$p(\boldsymbol{x}, t) = p_{\text{vapor}}(t) - \gamma\kappa + 2\mu_{\text{liquid}}(D_{\text{liquid}} \cdot \boldsymbol{n}) \cdot \boldsymbol{n},$$

where κ is the local mean curvature. $\boldsymbol{H}$ represents the force due to gravity $\boldsymbol{H} = (0, g)$, μ_{liquid} is the liquid viscosity (which we shall assume is zero), and D_{liquid} is the rate of deformation tensor for liquid.

3. CLSVOF free surface representation

The free surface is represented through a "coupled level set and volume-of-fluid" (CLSVOF) method. In addition to solving the level set equation (4), we shall also solve the following equation for the volume-of-fluid function F:

$$F_t + \boldsymbol{U} \cdot \nabla F = 0.$$

At $t = 0$, F is initialized in each computational cell Ω_{ij},

$$\Omega_{ij} = \{(x, y) \,|\, x_i \leqslant x \leqslant x_{i+1} \text{ and } y_j \leqslant y \leqslant y_{j+1}\},$$

to be,

$$F_{ij} = \frac{1}{\Delta x \Delta y} \int_{\Omega_{ij}} H(\phi(x, y, 0))\, \mathrm{d}x\, \mathrm{d}y.$$

Here, Δx and Δy are defined as $x_{i+1} - x_i$ and $y_{i+1} - y_i$, respectively, and $H(\phi)$ is the Heaviside function,

$$H(\phi) = \begin{cases} 1 & \phi \geqslant 0, \\ 0 & \text{otherwise.} \end{cases}$$

The discrete level set function $\phi^n_{i,j}$ and discrete volume fraction function $F^n_{i,j}$ are located at cell centers (see Fig. 3). The motion of the free surface is determined by the MAC (face-centered) velocities derived from the momentum equation. The discrete MAC velocity field is defined at cell faces $u_{i+1/2,j}$ and $v_{i,j+1/2}$, and satisfies the discrete continuity condition at every point in the liquid,

$$D^{\text{MAC}}\boldsymbol{U} = \frac{u_{i+(1/2),j} - u_{i-(1/2),j}}{\Delta x} + \frac{v_{i,j+(1/2)} - v_{i,j-(1/2)}}{\Delta y} = 0.$$

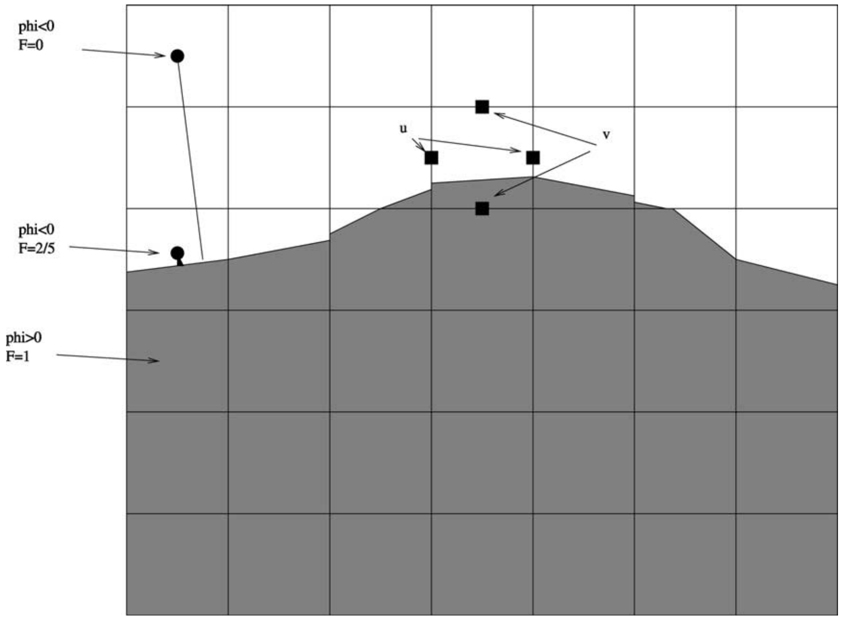

Fig. 3. Location of level set function ϕ, volume-of-fluid function F and MAC velocities in relation to the computational grid.

Details for the update of the level set function ϕ and the volume fractions F are given in [30] and in Appendix A below.

The level set function ϕ and the volume fractions F are coupled as follows:

- The normals used in the volume-of-fluid reconstruction step are determined from the level set function.
- The level set function is used to "truncate" the volume fractions; the truncation step removes spurious volumes ("flotsam"), generally created by round off error, that exist more than one grid cell length from the zero level set.
- The volume fractions are used, together with the slopes from the level set function, to construct a "volume-preserving" distance function along with providing "closest point" information to the zero level set.
- The volume fractions are used to express the interfacial curvature to second order accuracy (see Section 5.5). We do not use the level set function for finding the curvature because our level set reinitialization step is only second order accurate; the curvature as computed from the level set function will not provide the second order accuracy that is provided directly from the volume fractions.

We remark, that there are possible alternative, higher order, representations of the interface [12,15]. We point out though, that the accuracy of our computations are limited by the order of accuracy of our treatment of the free surface boundary conditions (which is second order); not by the accuracy of our interface representation. Our results in Section 7 bare this out. We demonstrate second order accuracy before and *after* pinch-off for complicated interfacial flows in which only first order methods have been applied previously. We also show that we conserve mass to a fraction of a percent for all of our computations.

4. Runge–Kutta time discretization for momentum equations

Our discretization procedure for approximating (1) is based on the "projection method" (see e.g. [5,6]). Given $\boldsymbol{U}^{\mathrm{MAC},n}$, ϕ^n, F^n and ∇p^n, we update using the following second order discretization:

1. Predict new position of the interface, $\phi^{n+1,(0)}$ and $F^{n+1,(0)}$, using $\boldsymbol{U}^{\mathrm{MAC},n}$ and the algorithm described in Section 3.
2. Runge–Kutta solve for predicting velocities:

$$\frac{\boldsymbol{U}^{*} - \mathscr{A}_{\mathrm{f}}^{\mathrm{c}}(\boldsymbol{U}^{\mathrm{MAC},n})}{\Delta t} = -L(\boldsymbol{U}^{\mathrm{MAC},n}) - \nabla p^{n} + \boldsymbol{H}^{n}, \tag{5}$$

   where L represents high order upwind conservative discretization of the nonlinear terms. The operator $\mathscr{A}_{\mathrm{f}}^{\mathrm{c}}$ represents the interpolation of the MAC, face-centered, velocity field to a cell-centered velocity field.
3. Projection step:

$$\begin{aligned} \boldsymbol{V} &\equiv \frac{\boldsymbol{U}^{*} - \mathscr{A}_{\mathrm{f}}^{\mathrm{c}}(\boldsymbol{U}^{\mathrm{MAC},n})}{\Delta t} + \nabla p^{n}, \\ \frac{\boldsymbol{U}^{\mathrm{MAC},n+1,(0)} - \boldsymbol{U}^{\mathrm{MAC},n}}{\Delta t} &= \mathscr{P}_{\phi^{n}}(\boldsymbol{V}), \\ \nabla p^{n+1,(0)} &= \boldsymbol{V} - \mathscr{A}_{\mathrm{f}}^{\mathrm{c}}\left(\frac{\boldsymbol{U}^{\mathrm{MAC},n+1,(0)} - \boldsymbol{U}^{\mathrm{MAC},n}}{\Delta t}\right). \end{aligned} \tag{6}$$

4. Update new position of the interface, ϕ^{n+1} and F^{n+1}, using $\frac{1}{2}(\boldsymbol{U}^{\mathrm{MAC},n} + \boldsymbol{U}^{\mathrm{MAC},n+1,(0)})$ and the algorithm described in Section 3.
5. Runge–Kutta solve for updated velocities:

$$\frac{\boldsymbol{U}^{*} - \mathscr{A}_{\mathrm{f}}^{\mathrm{c}}(\boldsymbol{U}^{\mathrm{MAC},n})}{\Delta t} = -L(\boldsymbol{U}^{\mathrm{MAC},n+1,(0)}) - \nabla p^{n+1,(0)} + \boldsymbol{H}^{n+1,(0)}.$$

6. Projection step:

$$\begin{aligned} \boldsymbol{V} &\equiv \frac{\boldsymbol{U}^{*} - \mathscr{A}_{\mathrm{f}}^{\mathrm{c}}(\boldsymbol{U}^{n})}{\Delta t} + \nabla p^{n+1,(0)}, \\ \frac{\boldsymbol{U}^{\mathrm{MAC},n+1,(1)} - \boldsymbol{U}^{\mathrm{MAC},n}}{\Delta t} &= \mathscr{P}_{\phi^{n+1,(0)}}(\boldsymbol{V}), \\ \nabla p^{n+1} &= \boldsymbol{V} - \mathscr{A}_{\mathrm{f}}^{\mathrm{c}}\left(\frac{\boldsymbol{U}^{\mathrm{MAC},n+1,(1)} - \boldsymbol{U}^{\mathrm{MAC},n}}{\Delta t}\right). \end{aligned} \tag{7}$$

7. Runge–Kutta step:

$$\boldsymbol{U}^{\mathrm{MAC},n+1} = \frac{1}{2}\left(\boldsymbol{U}^{\mathrm{MAC},n+1,(0)} + \boldsymbol{U}^{\mathrm{MAC},n+1,(1)}\right).$$

Remarks:
- The above method corresponds to a second order "TVD preserving" [27] Runge–Kutta method.
- The projection steps (6) and (7) decompose the cell-centered velocity field $\boldsymbol{V}$ into a divergence free face-centered vector field $\boldsymbol{V}_{\mathrm{d}}$ and gradient of a scalar ∇p:

$$\begin{aligned} \boldsymbol{V}_{\mathrm{d}} + \nabla p &= \mathscr{A}_{\mathrm{c}}^{\mathrm{f}}(\boldsymbol{V}), \\ \nabla \cdot \nabla p &= \nabla \cdot \mathscr{A}_{\mathrm{c}}^{\mathrm{f}}(\boldsymbol{V}), \\ \boldsymbol{V}_{\mathrm{d}} &= \mathscr{A}_{\mathrm{c}}^{\mathrm{f}}(\boldsymbol{V}) - \nabla p. \end{aligned} \tag{8}$$

The elliptic Eq. (8) is solved using the preconditioned conjugate gradient method. Second order (see Section 5.3 and [17]) Dirichlet pressure boundary conditions are given at the zero level set of ϕ.

5. Spatial discretization

We describe the discretization for the face/cell interpolation operators, $\mathscr{A}_{\mathrm{f}}^{\mathrm{c}}$ and $\mathscr{A}_{\mathrm{c}}^{\mathrm{f}}$, nonlinear terms, L (5), projection step $\mathscr{P}_\phi$ (6) and (7), second order, volume conserving, velocity extension from the liquid into the gas, and curvature.

5.1. Face/cell interpolation operators

The face to cell interpolation operator, $\mathscr{A}_{\mathrm{f}}^{\mathrm{c}}$, initializes a cell-centered velocity field from a specified MAC-located velocity field:

$$u_{ij} = \frac{u^{\mathrm{MAC}}_{i+1/2,j} + u^{\mathrm{MAC}}_{i-1/2,j}}{2},$$

$$v_{ij} = \frac{v^{\mathrm{MAC}}_{i,j+1/2} + v^{\mathrm{MAC}}_{i,j-1/2}}{2}.$$

The cell-to-face interpolation operator, $\mathscr{A}_{\mathrm{c}}^{\mathrm{f}}$, initializes a face-centered (MAC-located) velocity field from a specified cell-centered velocity field:

$$u^{\mathrm{MAC}}_{i+1/2,j} = \frac{1}{2}(u_{i,j} + u_{i+1,j}),$$

$$v^{\mathrm{MAC}}_{i,j+1/2} = \frac{1}{2}(v_{i,j} + v_{i,j+1}). \tag{9}$$

5.2. Discretization of nonlinear terms

The term,

$$\boldsymbol{F}_x + \boldsymbol{G}_y - (\nabla \cdot \boldsymbol{U})\boldsymbol{U}$$

is discretized as

$$L(\boldsymbol{U}^{\mathrm{MAC}})_{ij} = \frac{(F_{i+1/2,j} - F_{i-1/2,j})/\Delta x + (G_{i,j+1/2} - G_{i,j-1/2})/\Delta y - (D\boldsymbol{U}^{\mathrm{MAC}})_{ij}\boldsymbol{U}^{\mathrm{cell}}_{ij}}{1 - (\Delta t/2)(D\boldsymbol{U}^{\mathrm{MAC}})_{ij}}, \tag{10}$$

where $\boldsymbol{U}^{\mathrm{cell}} = \mathscr{A}_{\mathrm{f}}^{\mathrm{c}}(\boldsymbol{U}^{\mathrm{MAC}})$, $(D\boldsymbol{U}^{\mathrm{MAC}})_{ij}$ is the discrete divergence operator,

$$(D\boldsymbol{U}^{\mathrm{MAC}})_{ij} = \frac{u^{\mathrm{MAC}}_{i+1/2,j} - u^{\mathrm{MAC}}_{i-1/2,j}}{\Delta x} + \frac{v^{\mathrm{MAC}}_{i,j+1/2} - v^{\mathrm{MAC}}_{i,j-1/2}}{\Delta y}, \tag{11}$$

and,

$$F_{i+1/2,j} = \begin{pmatrix} u^{\mathrm{MAC}}_{i+1/2,j} u_{i+1/2,j} \\ u^{\mathrm{MAC}}_{i+1/2,j} v_{i+1/2,j} \end{pmatrix}, \quad G_{i,j+1/2} = \begin{pmatrix} v^{\mathrm{MAC}}_{i,j+1/2} u_{i,j+1/2} \\ v^{\mathrm{MAC}}_{i,j+1/2} v_{i,j+1/2} \end{pmatrix}. \tag{12}$$

The quantities $u_{i+1/2,j}$, $v_{i+1/2,j}$, $u_{i,j+1/2}$ and $v_{i,j+1/2}$ are constructed from $\boldsymbol{U}^{\text{cell}}$ using upwind and slope-limited differencing; e.g.,

$$u_{i,j+1/2} = \begin{cases} u_{ij}^{\text{cell}} + \frac{\Delta y}{2} u_{y,ij}^{\text{cell}} & \text{if } v_{\text{MAC},i,j+1/2} > 0, \\ u_{ij+1}^{\text{cell}} - \frac{\Delta y}{2} u_{y,ij+1}^{\text{cell}} & \text{if } v_{\text{MAC},i,j+1/2} < 0. \end{cases}$$

The slopes $u_{y,ij}^{\text{cell}}\Delta y$ are computed using second order Van Leer slope limiting [34],

$$u_{y,ij}^{\text{cell}}\Delta y = \begin{cases} S\min(2|u_{i,j+1}^{\text{cell}} - u_{i,j}^{\text{cell}}|, 2|u_{i,j}^{\text{cell}} - u_{i,j-1}^{\text{cell}}|, \frac{1}{2}|u_{i,j+1}^{\text{cell}} - u_{i,j-1}^{\text{cell}}|) & \text{if } s > 0, \\ 0 & \text{otherwise,} \end{cases}$$

where

$$S = \text{sign}(u_{i,j+1}^{\text{cell}} - u_{i,j-1}^{\text{cell}})$$

and

$$s = \left(u_{i,j+1}^{\text{cell}} - u_{i,j}^{\text{cell}}\right)\left(u_{i,j}^{\text{cell}} - u_{i,j-1}^{\text{cell}}\right).$$

Remark. The term $(D\boldsymbol{U}^{\text{MAC}})_{ij}$ appearing in (10) is discretely zero at every point in the liquid and also at "almost every" point in the extended liquid region. For extreme cases, e.g., a vapor region containing a single point, it is impossible to construct divergence free extension velocities at these points; necessitating the terms involving $(D\boldsymbol{U}^{\text{MAC}})_{ij}$. The $\Delta t/2(D\boldsymbol{U}^{\text{MAC}})_{ij}$ term in the denominator of (10) results from handling the latter part of $(\nabla\cdot\boldsymbol{U})\boldsymbol{U}$, $\boldsymbol{U}$, semi-implicitly.

5.3. Cell-centered projection step

The projection step is described analytically by (8). The discrete projection has the following steps:

1. We are given a discrete cell-centered vector field $\boldsymbol{V}_{ij}$.
2. Define $\boldsymbol{V}^{\text{MAC}} \equiv \mathscr{A}_{\text{c}}^{\text{f}}(\boldsymbol{V})$ see (9).
3. Solve the following discrete system using preconditioned conjugate gradient method:

$$DGp = DV^{\text{MAC}}, \tag{13}$$

D represents the discrete divergence operator,

$$DV^{\text{MAC}} = \frac{u_{i+1/2,j}^{\text{MAC}} - u_{i-1/2,j}^{\text{MAC}}}{\Delta x} + \frac{v_{i,j+1/2}^{\text{MAC}} - v_{i,j-1/2}^{\text{MAC}}}{\Delta y}$$

and $G = (G_x, G_y)$ represents the discrete gradient operator,

$$(G_x p)_{i+1/2,j} = \frac{p_{i+1,j} - p_{ij}}{\Delta x},$$

$$(G_y p)_{i,j+1/2} = \frac{p_{i,j+1} - p_{ij}}{\Delta y}.$$

In cells where the level set function changes sign, the gradient operator is modified in order to enforce second order Dirichlet boundary conditions at the vapor/liquid boundary. We treat the boundary condition similarly as was done in [17]. Please refer to Fig. 4. Suppose $\phi_{ij} < 0$ and $\phi_{i+1,j} > 0$; using linear interpolation, one can find the position of the zero level set to be $(x_{ij} + (1-\theta)\Delta x, y_j)$ where $0 < \theta < 1$. In order to

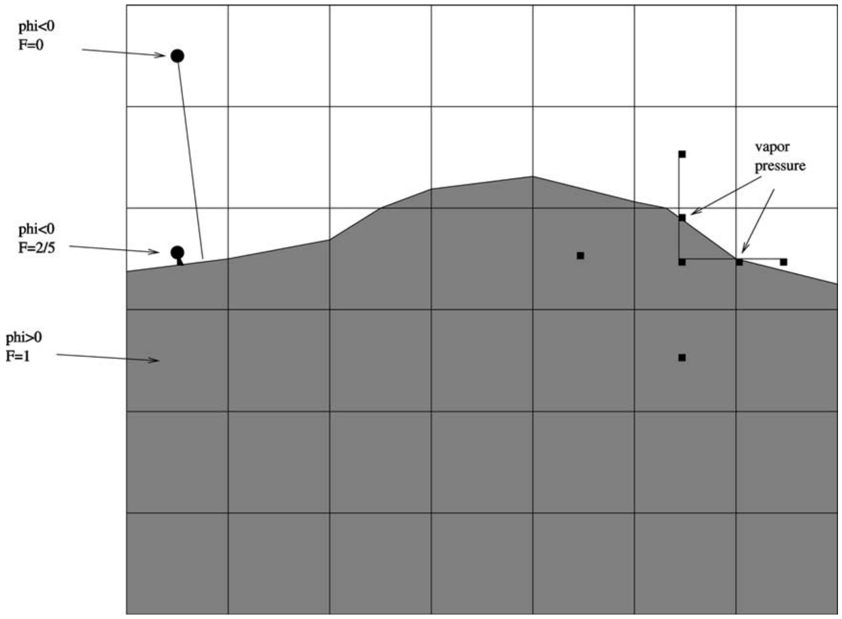

Fig. 4. Diagram of five-point stencil near an irregular boundary. The factor θ represents the fraction of the distance of the line connecting the center of the stencil to a stencil point outside of the liquid region.

avoid difficulties with numerical round-off, we place a lower bound on θ so that $10^{-3} < \theta < 1$. The resulting modification to $(G_x p)_{i+1/2,j}$ becomes

$$(G_x p)_{i+1/2,j} = \frac{p_{i+1,j} - p_v}{\theta \Delta x}.$$

The vapor pressure is

$$p_\mathrm{v} = p_\mathrm{vapor}(t) - \gamma \kappa_\mathrm{I}(F), \tag{14}$$

where the interface curvature κ_I is determined *directly* from the volume fractions F. The discretization of the curvature is described in Section 5.5.

4. Extend $\boldsymbol{u}^\mathrm{MAC}$ into the vapor; see Section 5.4.

Remark. Our algorithm is distinct from previous cell-centered, collocated, projection algorithms (see, e.g. [2]), since we project a cell-centered quantity but store the MAC velocities, $\boldsymbol{U}^{\mathrm{MAC},n}$, from time step to time step. This eliminates the ‘‘diffusive’’ problems encountered by repeatedly averaging the MAC velocities (see [22]) since we are able to (a) use $\boldsymbol{U}^{\mathrm{MAC},n}$ when discretizing the nonlinear terms (12) and (b) the errors associated with averaging the MAC velocities are not accumulated over time. In Results, we demonstrate single fluid computational results which are just as accurate as ‘‘node-based’’ projection algorithms. Our two-phase computations are shown to be second order accurate. In the future, the fact that we project the cell-centered quantity $\boldsymbol{V}$ (6) will make it convenient for us to include viscous effects implicitly using existing second order Crank–Nicolson solvers.

5.4. MAC velocity extension

We describe the extension of $\boldsymbol{u}^\mathrm{MAC}$ below. For a background of prior discretizations for this step, we refer the reader to [9–11,16]. Our method is distinct from existing methods for extending the MAC velocities in that it is second order and "volume conserving". The methods presented in [9,11,16] are all first order accurate; the SUMMAC method [10] describes a second order method for velocity extension, but their extension velocities are not discretely divergence free.

The extension algorithm consists of two steps: (1) extrapolation, (2) projection.

Step (1) is the extrapolation step where we initialize second order "non-volume-conserving" extension velocities for $\boldsymbol{u}^{\mathrm{MAC}}$. We describe the part that initializes $u^{\mathrm{MAC,extend}}_{i+1/2,j}$ below; the case for $v^{\mathrm{MAC,extend}}_{i,j+1/2}$ follows accordingly:

1. For each point where $0 > \phi_{i+1/2,j} \equiv (1/2)(\phi_{ij} + \phi_{i+1,j}) > -K\Delta x$, we already know the corresponding closest point on the interface $\boldsymbol{x}_{I,i+1/2,j} \equiv (1/2)(\boldsymbol{x}_{I,ij} + \boldsymbol{x}_{I,i+1,j})$. This information is derived during the CLSVOF reinitialization step (see Section A.1).
2. Construct a 5×5 stencil for u^{MAC} about the point $\boldsymbol{x}_{I,i+1/2,j}$. A point $\boldsymbol{x}_{i'+1/2,j'}$ in the stencil is tagged as "valid" if $\phi_{i',j'} \geqslant 0$ or $\phi_{i'+1,j'} \geqslant 0$. A diagram of how this 5×5 stencil is created for extending the horizontal velocity u^{MAC} is shown in Fig. 5. Please see Fig. 6 for a diagram portraying the 5×5 stencil used for constructing the vertical extension velocities $v^{\mathrm{MAC,extend}}_{i,j+1/2}$.
3. Find the weighted linear least squares best fit polynomial $u^{\mathrm{MAC,fit}}_{i+1/2,j} = a(x - x_{I,i+1/2,j}) + b(y - y_{I,i+1/2,j}) + c$ from the valid velocities in the 5×5 stencil. The weights are larger for those points near $\boldsymbol{x}_{I,i+1/2,j}$.
4. $$u^{\mathrm{MAC,extend}}_{i+1/2,j} = a(x_{i+1/2,j} - x_{I,i+1/2,j}) + b(y_{i+1/2,j} - y_{I,i+1/2,j}) + c.$$

Step (2) is the projection step where the newly extended velocity field $\boldsymbol{u}^{\mathrm{MAC,extend}}$ is "projected" onto a divergence free velocity field. We solve (13)

$$DGp^{\mathrm{extend}} = D\boldsymbol{u}^{\mathrm{MAC,extend}},$$

for those points where $-K\Delta x < \phi_{ij} < 0$. Homogeneous Neumann conditions are applied at faces where the level set function changes sign; e.g., $p^{\mathrm{extend}}_{i+1,j} = p^{\mathrm{extend}}_{i,j}$ if $\phi_{i+1,j} \geqslant 0$ and $\phi_{i,j} < 0$. Homogeneous Dirichlet

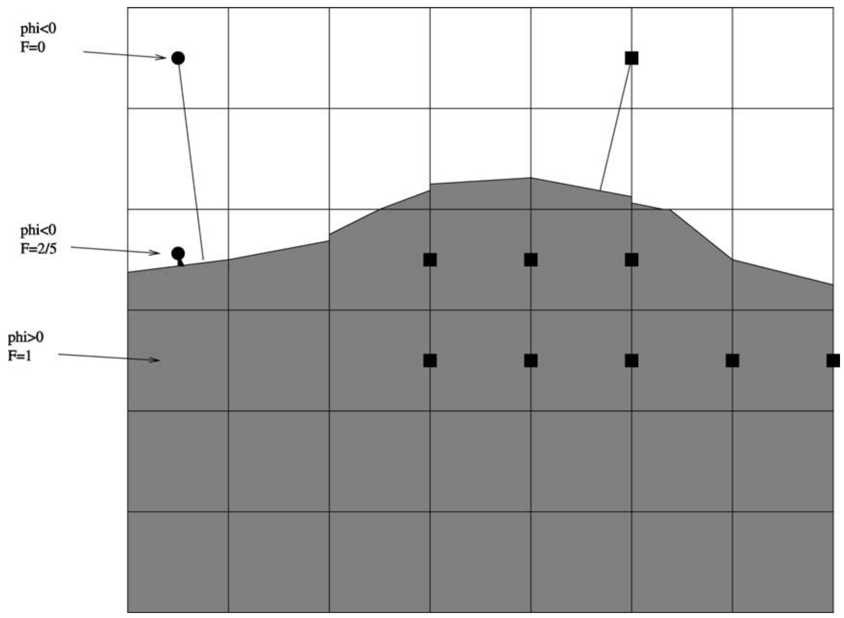

Fig. 5. Diagram highlighting the valid points in the 5 × 5 stencil used for constructing the horizontal extension velocities.

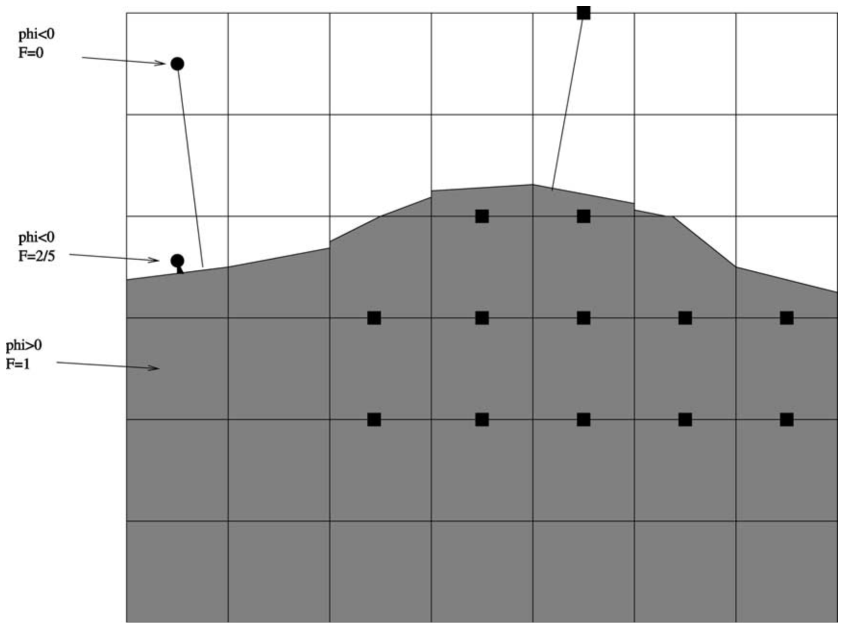

Fig. 6. Diagram highlighting the valid points in the 5×5 stencil used for constructing the vertical extension velocities.

boundary conditions are enforced at the $\phi = -K\Delta x$ level set. If $\phi > -K\Delta x$ throughout the vapor region, then homogeneous Dirichlet boundary conditions are enforced at the points where ϕ_{ij} is a local minimum (i.e., $\phi_{ij} \leqslant \phi_{i\pm1,j}$ and $\phi_{ij} \leqslant \phi_{i,j\pm1}$).

5.5. Curvature discretization

The curvature on the free surface is computed to second order accuracy directly from the volume fractions [19]. Previous work in this area include that by Chorin [13], Poo and Ashgriz [23], Aleinov and Puckett [1], Williams et al. [35] and more recently, using "PROST", Renardy et al. [25]. The method we use here is explicit, localized, and can be shown through Taylor series expansion to be second order accurate for $r - z$ or three-dimensional coordinate systems. The method is based on reconstructing the "height" function directly from the volume fractions [19]. Without loss of generality, we assume that the free surface is oriented more horizontal than vertical. The orientation of the free surface is determined from the level set function since $\boldsymbol{n} = \nabla\phi/|\nabla\phi|$. A 3×7 stencil of volume fractions is constructed about cell (i,j) (see Fig. 7). The three vertical sums, $F_{i'}$, $i' = i-1, i, i+1$ are exact integrals of the height function $h(x)$ (up to a constant); i.e., $F_i = \int_{x_{i-1/2}}^{x_{i+1/2}} h(x)\,\mathrm{d}x + C(j)$. It can be shown that $(F_{i+1} - F_{i-1})/\Delta x$ is a second order approximation to $h'(x_i)$ and that $(F_{i+1} - 2F_i + F_{i-1})/\Delta x^2$ is a second order approximation to $h''(x_i)$. A slightly more complicated procedure is used in axisymmetric coordinate systems; the height function $h(r)$ is assumed to have the form $ar^2 + br + c$. The integral of $rh(r)$ is related with $F_{i'}$, $i' = i-1, i, i+1$ in order to solve for the three unknowns a, b and c. For vertically oriented interfaces in axisymmetric coordinate systems, the $F_{j'}$ represent the integrals of the square of the height function $h(z)$ (up to a constant): $F_{j'} = \pi\int_{z_{j'-1/2}}^{z_{j'+1/2}} (h(z))^2\,\mathrm{d}z + C(i)$. In other words, $(F_{j+1} - F_{j-1})/\Delta x$ is a second order approximation to $\mathrm{d}h(z)^2/\mathrm{d}z$ and $(F_{j+1} - 2F_j + F_{j-1})/\Delta x^2$ is a second order approximation to $\mathrm{d}^2(h(z)^2)/\mathrm{d}z^2$. The resulting curvature is obtained directly from the height function (whether it be $h(r)$, $h(z)$ or $h(x,y)$).

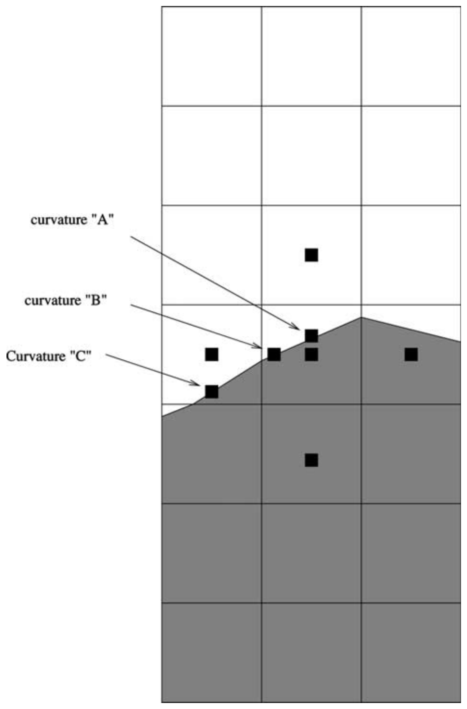

Fig. 7. The volume fractions in the following 3×7 stencil are used to approximate curvature "A" to second order accuracy. In order to compute curvature "B" to second order accuracy, one must linearly interpolate between curvature "A" and curvature "C".

This procedure for finding curvature will return a second order approximation to the curvature on the interface passing through cell (i,j) located at $x = (i + 1/2)\Delta x$ (horizontal orientation) or $y = (j + 1/2)\Delta y$ (vertical orientation). In order to find $\kappa_{\mathrm{I}}(F)$ to second order accuracy (14), we have two different cases when the level set function changes sign between cells (i,j) and $(i+1,j)$: (1) the interface is orientated vertically, in which case

$$\kappa_{\mathrm{I}} = \begin{cases} \kappa_{ij} & \theta < 1/2, \\ \kappa_{i+1,j} & \text{otherwise}, \end{cases}$$

or (2) the interface is orientated horizontally, in which case

$$\kappa_{\mathrm{I}} = (1-\theta)\kappa_{ij} + \theta\kappa_{i+1,j}.$$

In Tables 1 and 2, we display the average error and maximum error for the case of a sphere in axisymmetric and three-dimensional coordinate systems, respectively.

Table 1
Convergence study for computing curvatures from volume fractions of a unit sphere in axisymmetric geometry

Δx	Max. error	Avg. error
1/16	0.0104	0.0037
1/32	0.0024	0.0009
1/64	0.0006	0.0002

The physical domain size is 2×4. Δx is the mesh spacing which is $2/n_x$ where n_x is the number of cells in the x direction. For all our tests, $\Delta x = \Delta y$.

Table 2
Convergence study for computing curvatures from volume fractions of a unit sphere in three-dimensional geometry

Δx	Max. error	Avg. error
1/8	0.094	0.0125
1/16	0.050	0.0036
1/32	0.010	0.0009

The physical domain size is $4 \times 4 \times 4$. Δx is the mesh spacing which is $4/n_x$ where n_x is the number of cells in the x direction. For all our tests, $\Delta x = \Delta y = \Delta z$.

5.6. Time step

The time step Δt at time t^n is determined by restrictions due to the CFL condition, gravity, and surface tension [8,32]

$$\Delta t < \min_{i,j}\left(\sqrt{\frac{1}{4\pi\gamma}}\Delta x^{3/2}, \frac{2\Delta x}{|\boldsymbol{u}^n| + \sqrt{|\boldsymbol{u}^n|^2 + 4\mathscr{F}^n\Delta x}}\right),$$

where

$$\mathscr{F}^n = |-Gp^n + \boldsymbol{H}|.$$

The second time step constraint is justified through the following simplified analysis. If we consider the simplified equation:

$$u_t = F,$$

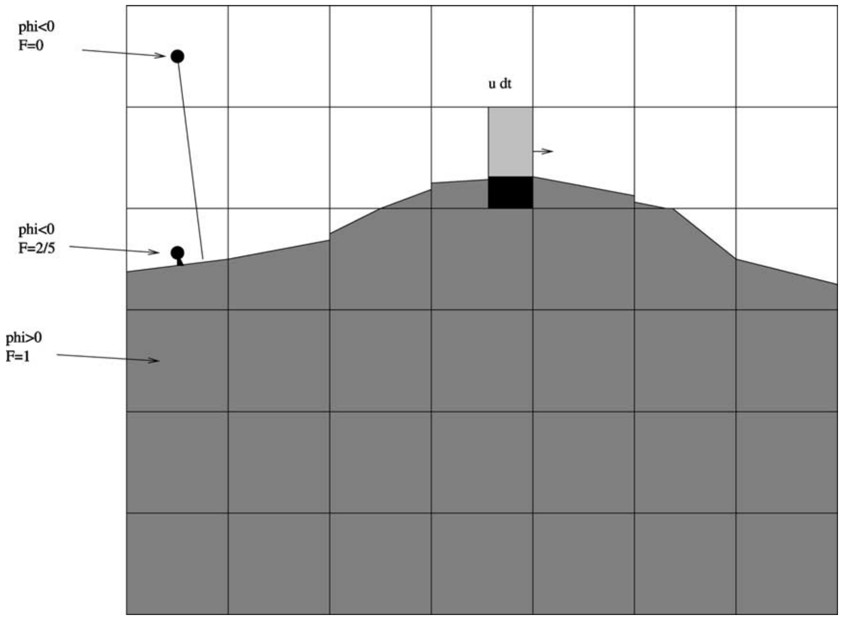

Fig. 8. The volume-of-fluid flux is represented by the fraction of total fluid being advected across the right face of the highlighted cell. The total amount of fluid being advected across the right face has volume $u\Delta t\Delta y$. The flux is represented by the fraction of "dark" material which is about 1/3 in this example.

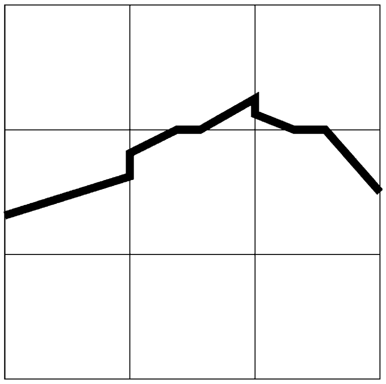
Fig. 9. The volume-of-fluid reconstructed interface consists of the piecewise linear line segments along with the portions of the cell boundaries that make up the difference between neighboring line segments. This is an exaggerated diagram. The magnitude of the discontinuity between neighboring segments is $O(\Delta x^2)$.

$$u(t_n) = u_n,$$

then the solution at t_{n+1} is

$$u(t_{n+1}) = u_n + \Delta t F.$$

We require a "CFL" type condition,

$$u(t_{n+1})\Delta t < \Delta x.$$

The resulting equation for Δt is

$$(u_n + \Delta t F)\Delta t < \Delta x.$$

6. Single fluid test

One obstacle to a cell-centered projection step is the "diffusive" problems encountered when one recovers the cell-centered velocity by averaging the projected MAC velocities [22]. In this work, we carry the MAC velocities from time step to time step in order to avoid these diffusive problems. We test the diffusiveness of our cell-centered projection algorithm on an inviscid horizontal shear layer problem [4]. The problem is computed in a doubly periodic 1×1 box with initial data given by:

$$u = \begin{cases} \tanh(30(y - 1/4)) & \text{for } y \leqslant 1/2, \\ \tanh(30(3/4 - y)) & \text{for } y > 1/2, \end{cases}$$

$$v = (1/20)\sin(2\pi x).$$

In Fig. 10, we display the vorticity contours for the 256×256 grid case. In Fig. 11, we plot the kinetic energy,

$$K = (1/2)\int \boldsymbol{u} \cdot \boldsymbol{u}\,\mathrm{d}x\,\mathrm{d}y$$

for the 256×256 grid case. These results are to be compared with those in [4]; we see that the error in kinetic energy for our results is only about 25% worse than [4].

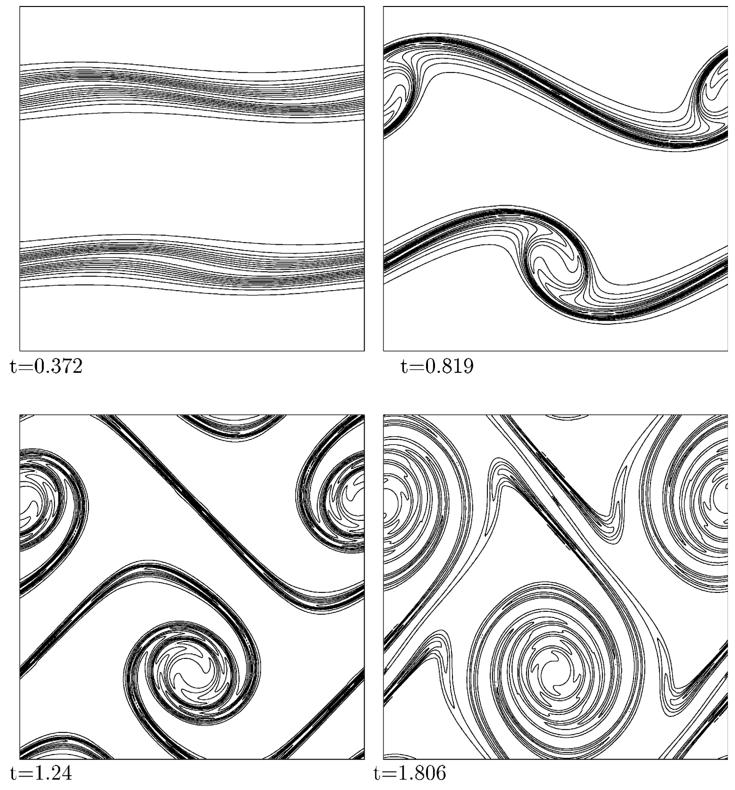

Fig. 10. Vorticity contours for smooth shear layer on 256 × 256 grid.

7. Two-phase flow tests

In this section we test the order of accuracy of our algorithm on two problems, (1) standing wave problem and (2) rising inviscid gas bubble problem. The first problem does not include surface tension effects whereas the second problem does. Since we do not know the exact solutions for these problems, we measure the relative error between succeeding grid resolutions. The error for the position of the free surface is measured as,

$$E(t) = \sum_{i,j} \int_{\Omega_{ij}} |H(\phi_{\mathrm{f}}(t)) - H(\phi_{\mathrm{c}}(t))|\, \mathrm{d}\boldsymbol{x}. \tag{15}$$

Here, ϕ_{c} is the level set function from a coarser computation and ϕ_{f} is the level set function from the refined computation. The relative error for the velocity is measured by the following equations:

$$E_{u,L1}(t) = \sum_{i,j,\phi_{\mathrm{c},ij}>0} \sqrt{(u_{\mathrm{f},ij} - u_{\mathrm{c},ij})^2 + (v_{\mathrm{f},ij} - v_{\mathrm{c},ij})^2}\Delta x \Delta y, \tag{16}$$

$$E_{u,\max}(t) = \max_{i,j,\phi_{\mathrm{c},ij}>0} \sqrt{(u_{\mathrm{f},ij} - u_{\mathrm{c},ij})^2 + (v_{\mathrm{f},ij} - v_{\mathrm{c},ij})^2}. \tag{17}$$

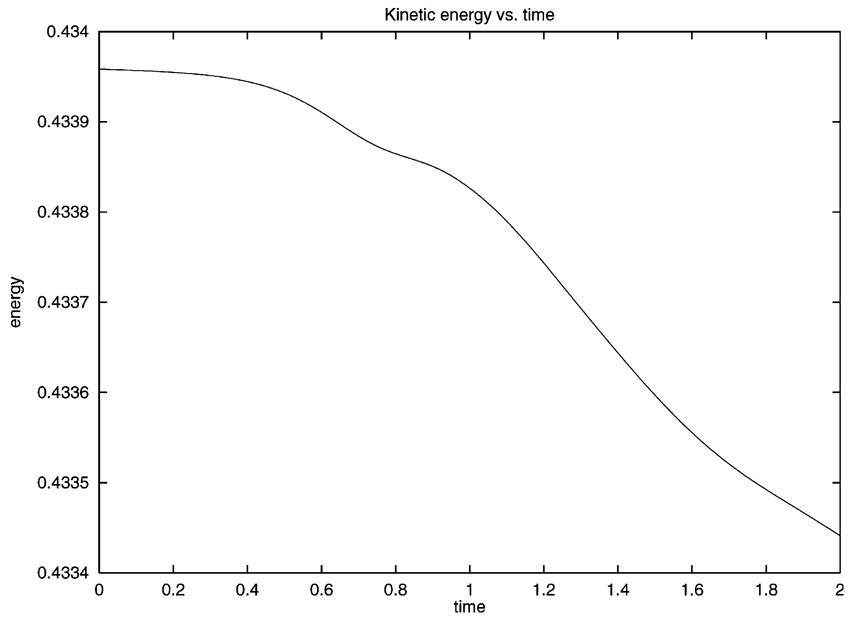

Fig. 11. Kinetic energy versus time for smooth shear layer; 256×256 grid case.

For the standing wave problem, the free surface at $t = 0$ is described by the equation

$$y = (1/4) + \epsilon\cos(2\pi x),$$

where $\epsilon = 0.025$. The gravitational force is $\boldsymbol{H} = (0, -2\pi)$. The computational domain is a 1/2 by 1/2 box with axisymmetric boundary conditions at $x = 0$ and $x = 1/2$ and solid wall boundary conditions at $y = 0$.

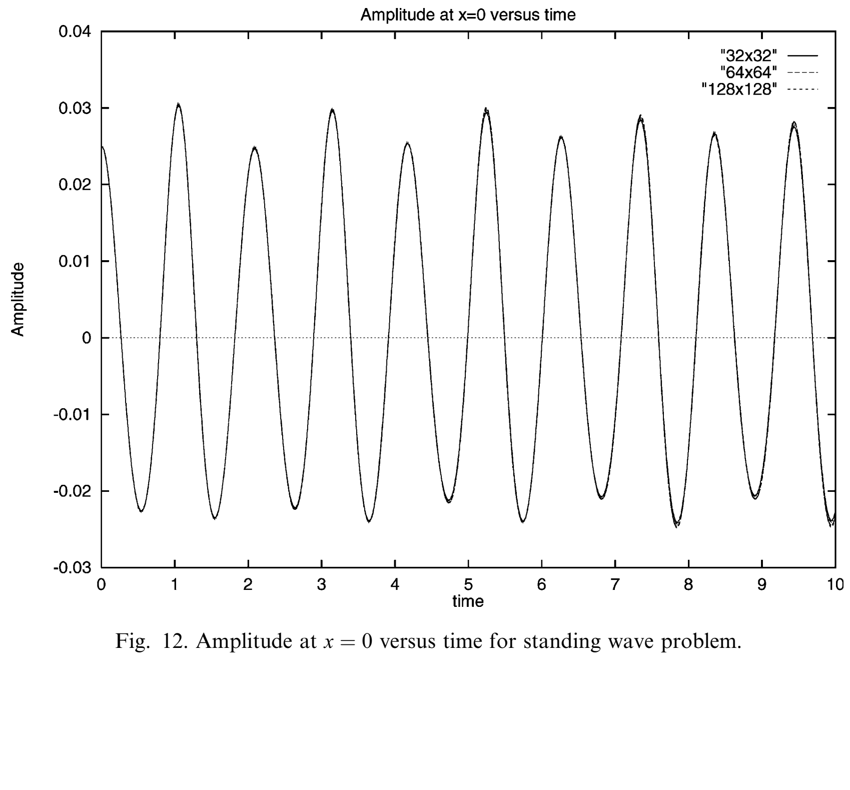

Fig. 12. Amplitude at $x = 0$ versus time for standing wave problem.

Table 3
Convergence study: relative error between coarse grid computations with cell size Δx_{coarse} and fine grid computations with cell size Δx_{fine} for amplitude at $x = 0$ for standing wave problem

Δx_{coarse}	Δx_{fine}	Max. relative error	Avg. relative error
1/64	1/128	6.7E−4	18.0E−5
1/128	1/256	2.2E−4	4.5E−5

The physical domain size is $1/2 \times 1/2$. Δx is the mesh spacing which is $1/2n_x$ where n_x is the number of cells in the x direction. For all our tests, $\Delta x = \Delta y$.

Table 4
Convergence study: relative error between coarse grid computations with cell size Δx_{coarse} and fine grid computations with cell size Δx_{fine} for velocity and free surface location at $t = 10$ for standing wave problem

Δx_{coarse}	Δx_{fine}	$E(10)$	$E_{u,L1}(10)$	$E_{u,\max}(10)$
1/64	1/128	11.0E−5	12.0E−5	3.7E−3
1/128	1/256	2.0E−5	3.1E−5	1.2E−3

The physical domain size is $1/2 \times 1/2$. Δx is the mesh spacing which is $1/2n_x$ where n_x is the number of cells in the x direction. For all our tests, $\Delta x = \Delta y$.

In Fig. 12 we compare the amplitude (at $x = 0$) for three different grid resolutions, $\Delta x = 1/64$, $\Delta x = 1/128$ and $\Delta x = 1/256$. The time step for each case is $\Delta t = 0.02$, $\Delta t = 0.01$ and $\Delta t = 0.005$. In Table 3, we show the relative error between the three graphs ($0 \leqslant t \leqslant 10$). In Table 4, we show the $E_{u,L1}$ and $E_{u,\max}$ errors for the velocity field at $t = 10.0$.

For the axisymmetric inviscid bubble rise problem, the free surface at $t = 0$ is a unit spherical bubble centered at $(0, 2)$ in a computational domain with dimensions of 3×6. The surface tension coefficient is $\gamma = 0.005$, and the gravitational force is $\boldsymbol{H} = (0, -1)$. These parameters correspond to those used for a convergence study by Sussman and Puckett [30]. The only difference between our problem setup and the setup in [30] is that we assume solid wall boundary conditions here as opposed to outflow boundary conditions. In Fig. 13 we display the results of our computations for the bubble rising as it breaks up. In Tables 5 and 6, we measure the relative errors for the interface and velocity field for grid resolutions ranging from 32×64 to 128×256 (Δt ranges from 0.005 to 0.00125). In Table 7, we show the maximum mass fluctuation for $0 \leqslant t \leqslant 1.75$. As shown in the tables, we obtain first order accuracy when measuring the error using $E_{u,\max}$ (prior to pinch-off) and second order accuracy when using $E_{u,L1}$; these errors are considerably lower than the results reported by Sussman and Puckett [30] using the "continuum" approach. In fact, the errors on the 128×256 grid using our recent algorithm correspond to those on the 256×512 grid used before; this corresponds to a factor of 8 speed-up.

8. Bubble oscillation and collapse

In this section, we test our algorithm on problems in which the pressure inside the vapor obeys the relation for an adiabatic perfect gas (3),

$$p_{\text{vapor}}(t) = p_0 \left(\frac{V(0)}{V(t)} \right)^{\gamma}, \tag{18}$$

where $V(t)$ is the volume of the vapor and $\gamma = 1.4$. In our computations, the volume is obtained directly from the volume fractions. At each time step, we identify the existing vapor bubble regions, $V_1(t), \ldots, V_N(t)$.

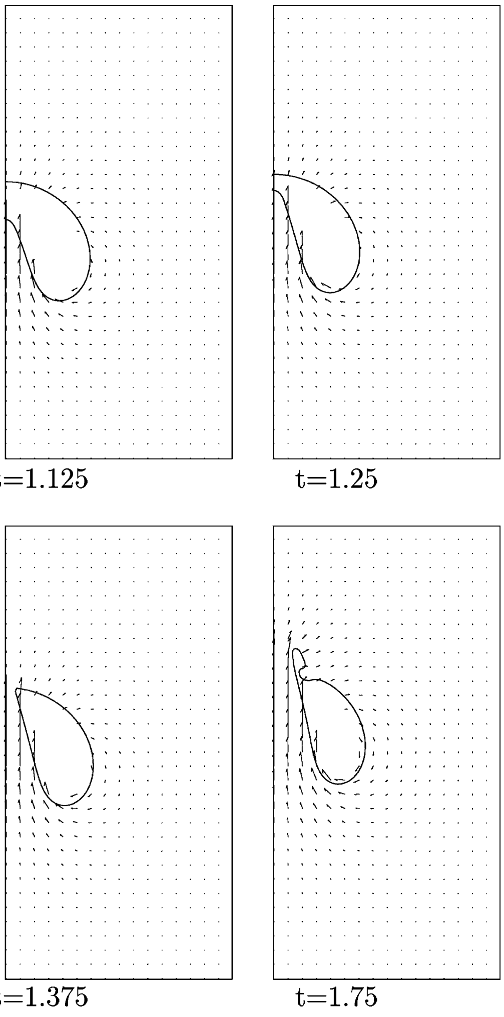

Fig. 13. Inviscid, axisymmetric gas bubble rising in liquid; 128 × 256 grid.

Table 5
Convergence study: relative error between coarse grid computations with cell size Δx_{coarse} and fine grid computations with cell size Δx_{fine} for velocity and free surface location at $t = 1.25$ for axisymmetric inviscid rising bubble problem

Δx_{coarse}	Δx_{fine}	$E(1.25)$	$E_{u,L1}(1.25)$	$E_{u,\max}(1.25)$
3/32	3/64	0.058	0.444	0.184
3/64	3/128	0.014	0.079	0.098

The physical domain size is 3 × 6. Δx is the mesh spacing which is $3/n_x$ where n_x is the number of cells in the x direction. For all our tests, $\Delta x = \Delta y$.

We also initialize the vapor pressure in each of these regions, and in a thin strip of one cell around each region. Computing the volume in each separate vapor region is a well defined operation since two computational cells can only belong to the same vapor region if they share either the same x coordinate or y coordinate, and the level set function does not change sign between the two.

Table 6
Convergence study: relative error between coarse grid computations with cell size Δx_{coarse} and fine grid computations with cell size Δx_{fine} for velocity and free surface location at $t = 1.375$ for axisymmetric inviscid rising bubble problem

Δx_{coarse}	Δx_{fine}	$E(1.375)$	$E_{u,L1}(1.375)$	$E_{u,\max}(1.375)$
3/32	3/64	0.082	0.530	0.170
3/64	3/128	0.017	0.095	0.211

The physical domain size is 3×6. Δx is the mesh spacing which is $3/n_x$ where n_x is the number of cells in the x direction. For all our tests, $\Delta x = \Delta y$.

Table 7
Maximum mass fluctuation of the bubble mass for axisymmetric inviscid rising bubble problem

Δx	Maximum mass fluctuation $0 \leqslant t \leqslant 1.75$
3/32	0.02%
3/64	0.02%
3/128	0.007%

The physical domain size is 3×6. Δx is the mesh spacing which is $3/n_x$ where n_x is the number of cells in the x direction. For all our tests, $\Delta x = \Delta y$.

In the event that a volume at a new time step, $V_j(t^{n+1})$, contains points from more than one region at the previous time step (i.e., a merger) or points from multiple volumes $V_j(t^{n+1})$ are contained in a single volume $V_k(t^n)$ at the previous time step (i.e., a break-up) then the pressure at the new time step is initialized as the average of the underlying vapor pressures that were defined at the previous time step; otherwise, (18) is used to initialize the new pressure using the volume and pressure at t^n and the new volume at t^{n+1}.

For our first test problem, we have a spherical, adiabatic vapor bubble surrounded by incompressible liquid. The initial radius of the bubble is $R(0) = 0.164$ and the initial pressure of the bubble is $p_0 = 100.0$. The pressure at the outer boundaries of the computational domain is $p_\infty = 1$. The computational domain

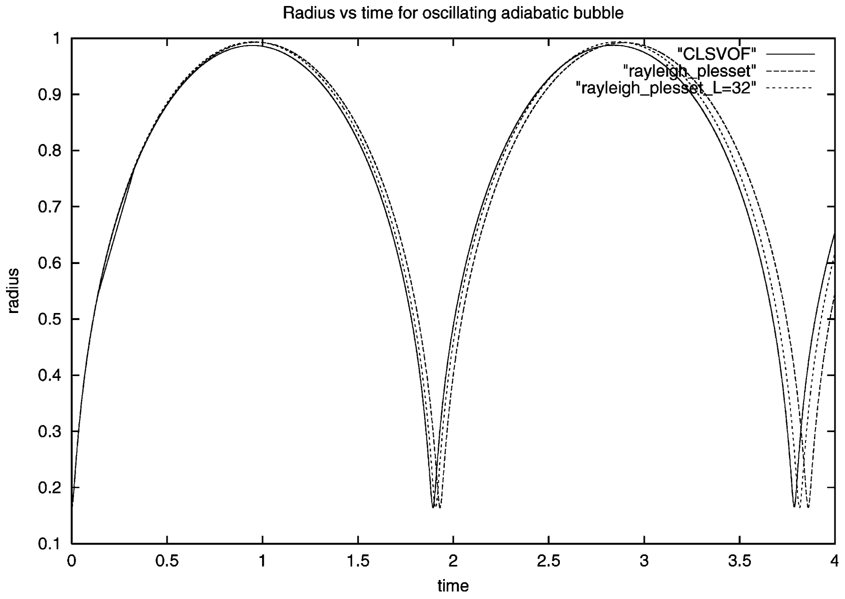

Fig. 14. Radius versus time for oscillating adiabatic bubble in an incompressible liquid. Coarse grid resolution 32×32. Domain size 32×32. Six levels of adaptivity used.

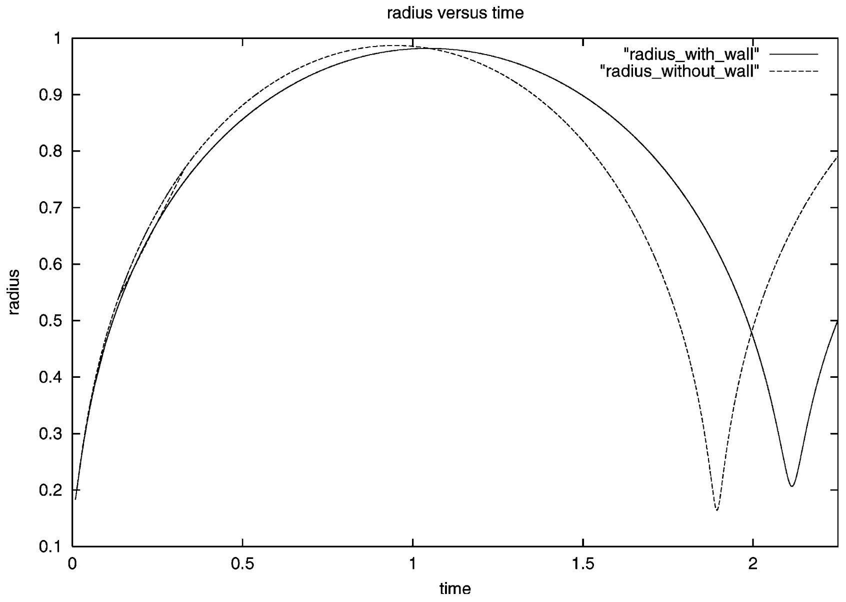

Fig. 15. Radius versus time for adiabatic bubble which grows and then collapses in an incompressible liquid. Coarse grid resolution 32×32. Domain size 32×32. Six levels of adaptivity used.

size is 32×32. We have symmetric boundary conditions at $r = 0,\ 0 \leqslant z \leqslant 32$ and $z = 0,\ 0 \leqslant r \leqslant 32$. The bubble center is at $r = 0, z = 0$. This is a standard test problem also done by Zhang et al. [36]. In our computation, we use adaptive mesh refinement [29] with a coarse grid size of 32×32 cells and a total of six levels of adaptivity. In Fig. 14, we compare the radius of the bubble from our computations to the radius computed from the Rayleigh–Plesset equation

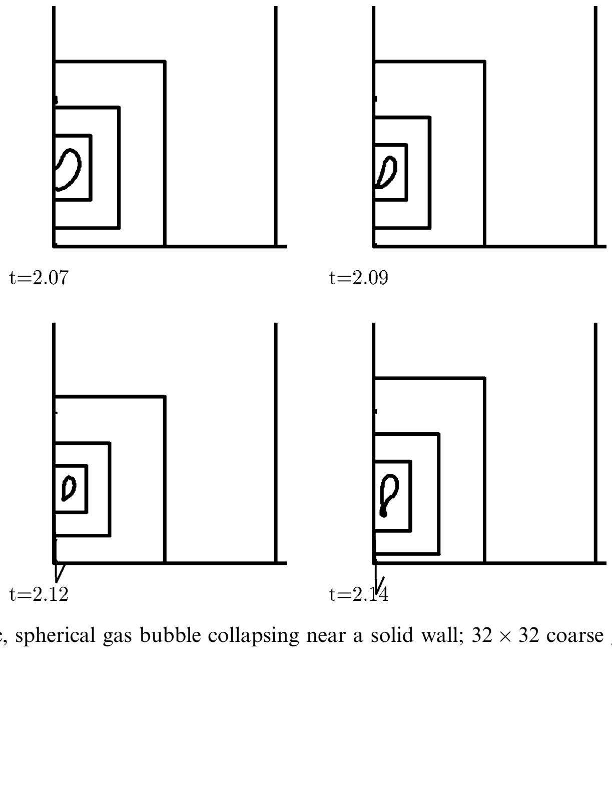

Fig. 16. Inviscid, adiabatic, spherical gas bubble collapsing near a solid wall; 32×32 coarse grid; 6 levels of adaptivity.

$$RR'' + (3/2)(R')^2 = p_0\left(\frac{R_0}{R}\right)^{3\gamma} - 1,$$

and the modified "finite domain" Rayleigh–Plesset equation

$$\frac{1}{1+R/L}RR'' + \left(\frac{1-R/3L}{1+R/L} + \frac{2(R/L)^4}{3(1+R/L)^4}\right)(3/2)(R')^2 = p_0\left(\frac{R_0}{R}\right)^{3\gamma} - 1,$$

$$L(R) = ((L_0 + R(0))^3 - R_0^3 + R^3)^{1/3} - R,$$

where $L_0 = 32$. The results between our computations and the finite domain Rayleigh–Plesset equation are almost identical.

For our second test problem, we again have a spherical, adiabatic vapor bubble surrounded by incompressible liquid. In this test problem, we place a solid boundary a distance of 1.5 units from the

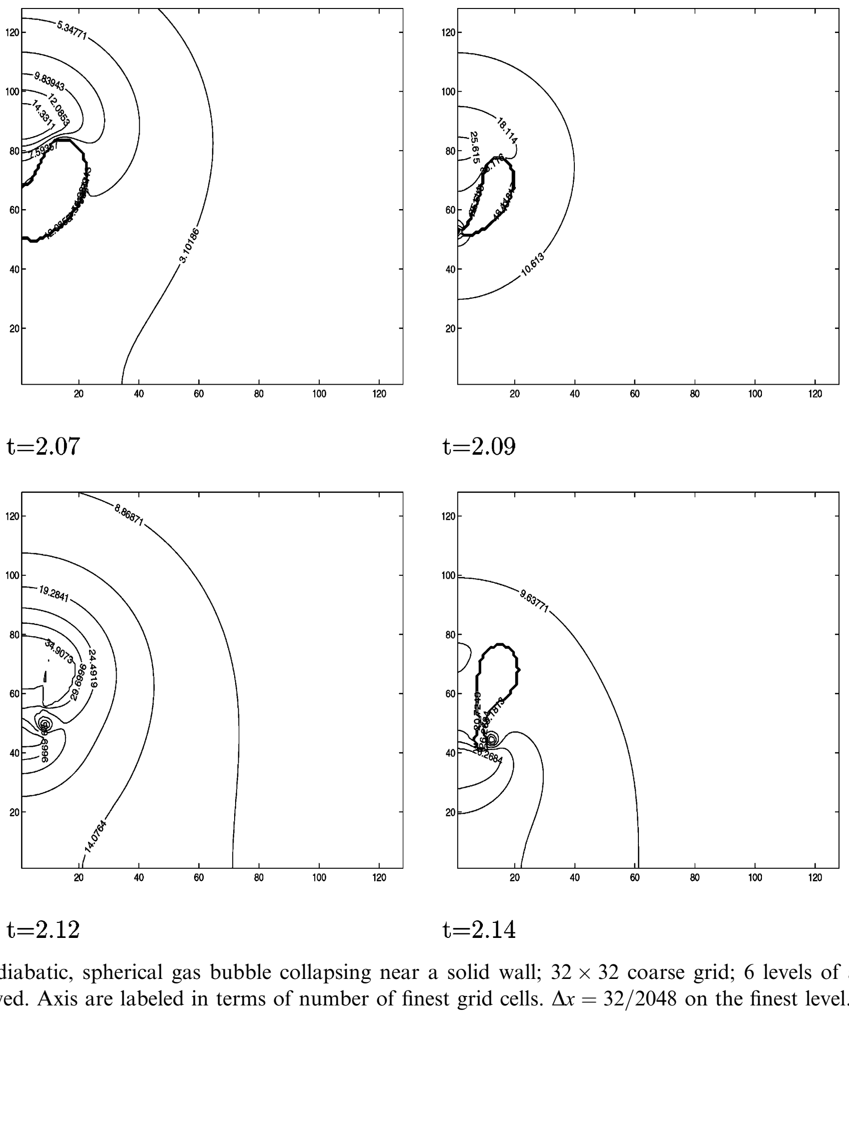

Fig. 17. Inviscid, adiabatic, spherical gas bubble collapsing near a solid wall; 32 × 32 coarse grid; 6 levels of adaptivity. Pressure contours are displayed. Axis are labeled in terms of number of finest grid cells. $\Delta x = 32/2048$ on the finest level.

center of the bubble. The bubble will grow, and then collapse, a liquid jet forms which impinges upon the solid boundary. The initial radius of the bubble is $R(0) = 0.164$ and the initial pressure of the bubble is $p_0 = 100.0$. The pressure at the outer boundaries of the computational domain is $p_\infty = 1$. The computational domain size is 32×32. We have symmetric boundary conditions at $r = 0,\ 0 \leqslant z \leqslant 32$ and solid wall boundary conditions at $z = 0,\ 0 \leqslant r \leqslant 32$. The bubble center is at $r = 0, z = 1.5$. This is a test problem also done by Zhang et al. [36]. In our computation, we use adaptive mesh refinement [29] with a coarse grid size of 32×32 cells and a total of six levels of adaptivity. In Fig. 15, we plot the radius of the bubble as it expands and then collapses. The radius is measured in our computations from the volume of the bubble. In Figs. 16 and 17 we show the pressure and interface profiles respectively as the bubble collapses.

9. Conclusions

Our results demonstrate second order accuracy measured using the L_1 norm for free surface flows with surface tension. The improved accuracy over conventional first order "continuum" approaches allows us to resolve computations using a grid resolution at least half the grid resolution as otherwise; this results in a 8:1 speedup for two-dimensional problems and a 16:1 speedup for three-dimensional problems.

Acknowledgements

Work supported in part by the National Science Foundation under contracts DMS 97-06847 and DMS 0108672, XEROX corporation, and ONR N00014-01-M-0238.

Appendix A

A.1. The CLSVOF advection algorithm

In this section, we describe how to advance the free surface using the coupled level set volume-of-fluid (CLSVOF) advection algorithm. We shall describe the details for the two-dimensional case. The three-dimensional algorithm follows analogously. We shall discretize our variables on a uniform grid with grid spacing of $\Delta x = \Delta y$. The discrete level set function $\phi^n_{i,j}$ and discrete volume fraction $F^n_{i,j}$ are located at cell centers. The motion of the free surface is determined by the velocity field derived from the equations for incompressible two-phase flow. The discrete velocity field is defined at cell faces $u_{i+1/2,j}$ and $v_{i,j+1/2}$, and satisfies the discrete divergence free condition

$$D^{\mathrm{MAC}}\boldsymbol{U} = \frac{u_{i+(1/2),j} - u_{i-(1/2),j}}{\Delta x} + \frac{v_{i,j+(1/2)} - v_{i,j-(1/2)}}{\Delta y}.$$

A diagram of where the discrete variables are located in relation to the computational grid is shown in Fig. 1. J represents the index of the computational cell closest to the top physical boundary.

The equations governing the interface motion are

$$\phi_t + \nabla \cdot (\boldsymbol{U}\phi) - (\nabla \cdot \boldsymbol{U})\phi = 0 \tag{A.1}$$

and

$$F_t + \nabla \cdot (\boldsymbol{U}F) - (\nabla \cdot \boldsymbol{U})F = 0.$$

We shall assume that the level set function $\phi^0_{i,j}$ is initialized as the signed normal distance from the initial position of the free surface. The volume fraction function $F^0_{i,j}$ shall be initialized as the fraction of liquid fluid contained in cell (i,j). In other words,

$$F^0_{i,j} = \frac{1}{\Delta x \Delta y}\int_{\Omega_{ij}} H(\phi(x,y,0))\,\mathrm{d}x\,\mathrm{d}y. \tag{A.2}$$

where

$$\Omega_{ij} = \left\{(x,y)\,|\,x_i \leqslant x \leqslant x_{i+1} \text{ and } y_j \leqslant y \leqslant y_{j+1}\right\}. \tag{A.3}$$

Given $\phi^n_{i,j}$, $F^n_{i,j}$ and $\boldsymbol{U}$, we use a "coupled" second order conservative operator split advection scheme in order to find $\phi^{n+1}_{i,j}$ and $F^{n+1}_{i,j}$. Second order accuracy is achieved by reversing the order of the splitting at each time step. In other words, for one time step, one solves (A.1) in the x direction first, and in the y direction next. For the next time step, the order is reversed, where one first solves in the y direction and then in the x direction. This operator splitting algorithm has been shown to yield a second order accurate advection algorithm [24,30].

The operator split algorithm for a general scalar s follows as:

$$\tilde{s}_{i,j} = \frac{s^n_{i,j} + (\Delta t/\Delta x)(G_{i-1/2,j} - G_{i+1/2,j})}{1 - (\Delta t/\Delta x)(u_{i+1/2,j} - u_{i-1/2,j})}, \tag{A.4}$$

$$s^{n+1}_{i,j} = \tilde{s}_{i,j} + \frac{\Delta t}{\Delta y}(\tilde{G}_{i,j-1/2} - \tilde{G}_{i,j+1/2}) + \tilde{s}_{i,j}(v_{i,j+1/2} - v_{i,j-1/2}), \tag{A.5}$$

where $G_{i+1/2,j} = s_{i+1/2,j}u_{i+(1/2),j}$ denotes the flux of s across the right face of the (i,j)th cell and $\tilde{G}_{i,j+1/2} = \tilde{s}_{i,j+1/2}v_{i,j+1/2}$ denotes the flux across the top face of the (i,j)th cell.

The scalar flux $s_{i+1/2,j}$ is computed differently depending on whether s represents the level set function ϕ or the volume fraction function F.

Computation of fluxes when s represents the level set function. For the case when s represents the level set function ϕ we have the following representation for $s_{i+1/2,j}$ when $u_{i+1/2,j} > 0$:

$$s_{i+1/2,j} = s^n_{i,j} + \frac{\Delta x}{2}\left(1 - u_{i+1/2,j}\frac{\Delta t}{\Delta x}\right)\frac{s^n_{i+1,j} - s^n_{i-1,j}}{\Delta x} \tag{A.6}$$

and when $u_{i+1/2,j} < 0$,

$$s_{i+1/2,j} = s^n_{i+1,j} - \frac{\Delta x}{2}\left(1 + u_{i+1/2,j}\frac{\Delta t}{\Delta x}\right)\frac{s^n_{i+2,j} - s^n_{i,j}}{\Delta x}.$$

The above discretization is motivated by the predictor corrector method described in [5] and the references therein. The scalar flux $s_{i+1/2,j}$ is obtained by extrapolating s in both space and time. Below, we show an example for the case when $u_{i+1/2,j} > 0$,

$$s_{i+1/2,j} \approx s_{i,j} + \frac{\Delta x}{2}s_{x,ij} + \frac{\Delta t}{2}s_{t,ij}. \tag{A.7}$$

For an operator split algorithm we only solve for one direction at a time. This means, for example, that we are solving

$$s_t + us_x = 0.$$

We can substitute $s_{t,ij} = -us_{x,ij}$ into (A.7) in order to obtain,

$$s_{i+1/2,j} \approx s_{i,j} + \frac{\Delta x}{2}\left(1 - u\frac{\Delta t}{\Delta x}\right)s_{x,ij}.$$

If we replace u with $u_{i+1/2,j}$ and $s_{x,ij}$ with $(s^n_{i+1,j} - s^n_{i-1,j})/\mathrm{d}x$, then we recover (A.6).

Computation of fluxes when s represents the volume-of-fluid function. For the case when s represents the volume-of-fluid function F we have the following representation for $s_{i+1/2,j}$ when $u_{i+1/2,j} > 0$:

$$s_{i+1/2,j} = \frac{\left(\int_{y_{j-1/2}}^{y_{j+1/2}} \int_{x_{i+1/2}-u_{i+1/2,j}\Delta t}^{x_{i+1/2}} H(\phi^{n,R}_{i,j}(x,y))\,\mathrm{d}x\,\mathrm{d}y\right)}{u_{i+1/2,j}\Delta t\Delta y} \tag{A.8}$$

and when $u_{i+1/2,j} < 0$,

$$\frac{\left(\int_{y_{i-1/2}}^{y_{i+1/2}} \int_{x_{i+1/2}}^{x_{i+1/2}-u_{i+1/2,j}\Delta t} H(\phi^{n,R}_{i+1,j}(x,y))\,\mathrm{d}x\,\mathrm{d}y\right)}{|u_{i+1/2,j}|\Delta t\Delta y}. \tag{A.9}$$

The volume-of-fluid flux $s_{i+1/2,j}$ represents the fraction of total fluid being advected across the right cell face that is “dark” fluid; see Fig. 8. The term $\phi^{n,R}_{i,j}(x,y)$ found in (A.8) and (A.9) represents the linear reconstruction of the interface in cell (i,j). In other words, $\phi^{n,R}_{i,j}(x,y)$ has the form

$$\phi^{n,R}_{i,j}(x,y) = a_{i,j}(x - x_i) + b_{i,j}(y - y_j) + c_{i,j}. \tag{A.10}$$

The coefficients $a_{i,j}$, $b_{i,j}$ and $c_{i,j}$ are first chosen so that (A.10) represents the best fit line for the piece of the zero level set passing through cell (i,j). In other words, a, b and c minimize the following error:

$$E_{i,j} = \int_{x_{i-1/2}}^{x_{i+1/2}} \int_{y_{i-1/2}}^{y_{i+1/2}} H'(\phi)(\phi - a_{i,j}(x - x_i) - b_{i,j}(y - y_j) - y_{i,j})^2. \tag{A.11}$$

In order to solve for a, b, and c, we minimize the discretized error,

$$E^{\Delta x}_{i,j} = \sum_{i'=i-1}^{i+1} \sum_{j'=j-1}^{j+1} w_{i'-i,j'-j} H'_\epsilon(\phi_{i',j'})(\phi_{i',j'} - a_{i,j}(x_{i'} - x_i) - b_{i,j}(y_{j'} - y_j) - c_{i,j})^2. \tag{A.12}$$

The discrete weights $w_{r,s}$ are chosen so that (A.12) is an approximation to (A.11). For the computations we show, we have $w_{r,s} = 16$ for $r = s = 0$ and $w_{r,s} = 1$ for $r \neq 0$ or $s \neq 0$. We have tried other values for $w_{r,s}$ with little effect on the accuracy of the computation. $H'_\epsilon(\phi)$ represents the smoothed Heaviside function with thickness ϵ; in our computations, we always have $\epsilon = \sqrt{2}\Delta x$. The resulting equations for a, b, c as a result of minimizing (A.12) is a 3×3 linear system.

The intercept $c_{i,j}$ is corrected so that the line represented by (A.10) cuts out the same volume in cell (i,j) as specified by $F^n_{i,j}$. In other words, the following equation is solved for $c_{i,j}$:

$$\int_{y_{i-1/2}}^{y_{i+1/2}} \int_{x_{i-1/2}}^{x_{i+1/2}} H(a_{i,j}(x - x_i) + b_{i,j}(y - y_j) + c_{i,j})\,\mathrm{d}y\,\mathrm{d}x = F^n_{i,j}. \tag{A.13}$$

Since H is a Heaviside function defined as $H(\phi) = 1$ if $\phi > 0$ and $H(\phi) = 0$ otherwise, we solve (A.13) by use of the Newton iteration method. We remark that the algorithm is simplified by first rotating the grid axis so that the normal represented by $a_{i,j}$ and $b_{i,j}$ points away from the lower left hand corner of the (i,j) computational cell. The coefficients $a_{i,j}$, $b_{i,j}$ and $c_{i,j}$ are also rescaled so that $a^2_{i,j} + b^2_{i,j} = 1$ and the new intercept represents the normal distance to the lower left hand corner of the computational cell.

The integrals in (A.8) and (A.9) are evaluated by finding the volume cut out of the region of integration by the line represented by (A.10).

The scalar flux $\tilde{s}_{i,j+1/2}$ is computed in the same manner as $s_{i+1/2,j}$. For the case when s represents the level set function ϕ, we have the following representation for $\tilde{s}_{i,j+1/2}$ when $v_{i,j+1/2} > 0$:

$$\tilde{s}_{i,j+1/2} = \tilde{s}_{i,j} + \frac{\Delta y}{2}\left(1 - v_{i,j+1/2}\frac{\Delta t}{\Delta y}\right)\frac{\tilde{s}_{i,j+1} - \tilde{s}_{i,j-1}}{\Delta y} \tag{A.14}$$

and when $v_{i,j+1/2} < 0$,

$$\tilde{s}_{i,j+1/2} = \tilde{s}_{i+1,j} - \frac{\Delta y}{2}\left(1 + v_{i,j+1/2}\frac{\Delta t}{\Delta y}\right)\frac{\tilde{s}_{i,j+2} - \tilde{s}_{i,j}}{\Delta y}. \tag{A.15}$$

For the case when $\tilde{s}$ represents the volume-of-fluid function F we have the following representation for $\tilde{s}_{i,j+1/2}$ when $v_{i,j+1/2} > 0$:

$$\tilde{s}_{i,j+1/2} = \frac{\left(\int_{y_{i+1/2}-v_{i,j+1/2}\Delta t}^{y_{i+1/2}} \int_{x_{i-1/2}}^{x_{i+1/2}} H(\tilde{\phi}^R_{i,j}(x,y))\,\mathrm{d}x\,\mathrm{d}y\right)}{v_{i,j+1/2}\Delta t\Delta x} \tag{A.16}$$

and when $v_{i,j+1/2} < 0$,

$$\tilde{s}_{i,j+1/2} = \frac{\left(\int_{y_{i+1/2}-v_{i,j+1/2}\Delta t}^{y_{i+1/2}} \int_{x_{i-1/2}}^{x_{i+1/2}} H(\tilde{\phi}^R_{i,j}(x,y))\,\mathrm{d}x\,\mathrm{d}y\right)}{|v_{i,j+1/2}|\Delta t\Delta x} \tag{A.17}$$

The linear reconstruction $\tilde{\phi}^R_{i,j}(x,y)$ found in (A.16) and (A.17) has an analogous form as (A.10),

$$\tilde{\phi}^R_{i,j}(x,y) = \tilde{a}_{i,j}(x - x_i) + \tilde{b}_{i,j}(y - y_j) + \tilde{c}_{i,j}. \tag{A.18}$$

After ϕ^{n+1} and F^{n+1} have been updated according to (A.4) and (A.5) we "couple" the level set function to the volume fractions by assigning the level set function ϕ^{n+1} to be the *exact* signed normal distance to the reconstructed interface. The reconstructed interface consists of the piecewise linear line segments defined by (A.18) along with the portions of the cell boundaries that make up the difference between neighboring line segments (see Fig. 9). The algorithm to find the signed normal distance in a strip of K cells about the reconstructed interface is as follows:

1. Truncate the volume fractions:

$$F^{n+1}_{i,j} = \begin{cases} 0 & \text{if } F^{n+1}_{i,j} \leqslant 0 \text{ or } \phi^{n+1}_{i,j} < -\Delta x, \\ 1 & \text{if } F^{n+1}_{i,j} \geqslant 1 \text{ or } \phi^{n+1}_{i,j} > \Delta x, \\ F^{n+1}_{i,j} & \text{otherwise.} \end{cases} \tag{A.19}$$

2. Tag all computational cells (i,j).
3. In each computational cell (i,j), check to see if

$$\phi^{n+1}_{i,j}\phi^{n+1}_{i',j'} \leqslant 0 \tag{A.20}$$

for some $|i - i'| \leqslant 1$, $|j - j'| \leqslant 1$; if there is a (i',j') such that (A.20) is satisfied, then perform the following steps:

(a) if

$$0 < F^{n+1}_{i,j} < 1 \tag{A.21}$$

and

$$\phi_{i,j}^{n+1}(\phi_{i,j}^{n+1} + \phi_{i',j'}^{n+1}) \leqslant 0 \quad \text{for some } |i - i'| \leqslant 1, |j - j'| \leqslant 1, \tag{A.22}$$

then construct the linear reconstruction $\phi_{i,j}^{n+1,R}(x,y)$ (A.10),

$$\phi_{i,j}^{n+1,R}(x,y) = a_{i,j}(x - x_i) + b_{i,j}(y - y_j) + c_{i,j}. \tag{A.23}$$

If (A.21) or (A.22) is not satisfied then mark all of cell (i,j) face centroids and corners as either "positive" or "negative" depending on the sign of $\phi_{i,j}^{n+1}$.

If both (A.21) and (A.22) are satisfied, mark all of cell (i,j) face centroids and corners according to the sign of $\phi_{i,j}^{n+1,R}(x,y)$ evaluated at the face centroids and corners.

(b) For each cell (i',j'), $(i'-i)^2 + (j'-j)^2 < K^2$ and $(i'-i)^2 + (j'-j)^2 < (|\phi_{i',j'}^{n+1}|/\Delta x + 2)^2$ do the following steps; we refer the reader to the diagram in Fig. 2.

(i) Determine the closest point on the boundary of cell (i,j) to $(x_{i'}, y_{j'})$ (this point will always either be at the corner or face centroid of the cell boundary). If the sign of the level set function at the closest point is *opposite* of $\phi_{i',j'}^{n+1}$, then set d, the shortest distance associated with cells (i,j) and (i',j'), equal to the distance from $(x_{i'}, y_{j'})$ to the closest point on the boundary of cell (i,j). If the sign of the level set function at the closest point is the *same* as $\phi_{i',j'}^{n+1}$ and (A.21) and (A.22) are both satisfied, then let d be the shortest distance between $(x_{i'}, y_{j'})$ and the line segment represented by $\phi_{i,j}^{n+1,R}(x,y)$.

(ii) Update $\phi_{i',j'}^{n+1}$ using d:

$$\phi_{i',j'}^{n+1} = \begin{cases} \operatorname{sign}(\phi_{i',j'}^{n+1})d & \text{if } d < |\phi_{i',j'}^{n+1}| \text{ or cell } (i',j') \text{ is tagged,} \\ \phi_{i',j'}^{n+1} & \text{otherwise.} \end{cases}$$

(iii) Untag cell (i',j').

4. For cells (i,j) which are still tagged, we have

$$\phi_{i,j}^{n+1} = \begin{cases} -K\Delta x - \Delta x & \text{if } \phi_{i,j}^{n+1} < 0, \\ K\Delta x + \Delta x & \text{if } \phi_{i,j}^{n+1} > 0. \end{cases} \tag{A.24}$$

Remarks:

- The coupling between the level set function ϕ and the volume-of-fluid function F occurs when computing the normal of the reconstructed interface (A.10) and also when assigning the level set function with the exact signed normal distance to the reconstructed interface.
- In order to find the shortest distance between the cell center (i',j') and the line segment represented by $\phi_{i,j}^{n+1,R}(x,y)$ (A.23), one first re-scales (A.23) so that $a_{i,j}^2 + b_{i,j}^2 = 1$. The distance is then $d = \phi_{i,j}^{n+1,R}(x_{i'}, y_{j'})$. The point $\boldsymbol{x}^c = (x_{i'}, y_{j'}) - d\nabla\phi^{n+1,R}$ is the point where the normal extension from (i',j') to $\phi_{i,j}^{n+1,R}(x,y)$ intersects $\phi_{i,j}^{n+1,R}(x,y)$. If $\boldsymbol{x}^c$ falls outside of cell (i,j), then the shortest distance between (i',j') and $\phi_{i,j}^{n+1,R}(x,y)$ must be the distance from (i',j') to one of the end points of the line *segment* represented by $\phi_{i,j}^{n+1,R}(x,y)$.

References

[1] I. Aleinov, E.G. Puckett, Computing surface tension with high-order kernels, in: Proceedings of the 6th International Symposium on Computational Fluid Dynamics, Lake Tahoe, CA, 1995.

[2] A.S. Almgren, J.B. Bell, P. Colella, T. Marthaler, A cell-centered cartesian grid projection method for the incompressible Euler equations in complex geometries, in: Proceedings of the 12th AIAA Computational Fluid Dynamics Conference, San Diego, CA, June 19–22 1995.

[3] S.W. Armfield, R. Street, Fractional step methods for the Navier–Stokes equations on non-staggered grids, ANZIAM J. 42 (E) (2000) C134–C156.
[4] J.B. Bell, P. Colella, H.M. Glaz, A second-order projection method for viscous, incompressible flow, in: Proceedings of the 8th AIAA Computational Fluid Dynamics Conference, Honolulu, June 9–11, 1987.
[5] J.B. Bell, P. Colella, H.M. Glaz, A second-order projection method for the incompressible Navier–Stokes equations, J. Comput. Phys. 85 (1989) 257–283.
[6] J.B. Bell, D.L. Marcus, A second-order projection method for variable-density flows, J. Comput. Phys. 101 (1992) 334–348.
[7] J. Best, The formation of toroidal bubbles upon the collapse of transient cavities, J. Fluid Mech. 251 (1993) 79–107.
[8] J.U. Brackbill, D.B. Kothe, C. Zemach, A continuum method for modeling surface tension, J. Comput. Phys. 100 (1992) 335–353.
[9] R. Caiden, R. Fedkiw, C. Anderson, A numerical method for two phase flow consisting of separate compressible and incompressible regions, J. Comput. Phys. 166 (2001) 1–27.
[10] R. Chan, R. Street, A computer study of finite-amplitude water waves, J. Comput. Phys. 6 (1970) 68–94.
[11] S. Chen, D. Johnson, P. Raad, Velocity boundary conditions for the simulation of free surface fluid flow, J. Comput. Phys. 116 (1995) 262–276.
[12] D. Chopp, Some improvements of the fast marching method, SIAM J. Sci. Comput. 23 (1) (2001) 230–244.
[13] A.J. Chorin, Curvature and solidification, J. Comput. Phys. 57 (1985) 472–490.
[14] N.V. Deshpande, Fluid mechanics of bubble growth and collapse in a thermal ink-jet printer, in: SPSE/SPIES Electronic Imaging Devices and Systems Symposium, January 1989.
[15] D. Enright, R. Fedkiw, R. Ferziger, I. Mitchell, A hybrid particle level set method for improved interface capturing, J. Comput. Phys. 183 (2002) 83–116.
[16] D. Enright, S. Marschner, R. Fedkiw, Animation and rendering of complex water surfaces, in: SIGGRAPH 2002, 2002, pp. 736–744.
[17] F. Gibou, R. Fedkiw, L.-T. Cheng, M. Kang, A second order accurate symmetric discretization of the poisson equation on irregular domains, J. Comput. Phys. 176 (2002) 205–227.
[18] B.T. Helenbrook, L. Martinelli, C.K. Law, A numerical method for solving incompressible flow problems with a surface of discontinuity, J. Comput. Phys. 148 (1999) 366–396.
[19] J. Helmsen, P. Colella, E.G. Puckett, Non-convex profile evolution in two dimensions using volume of fluids, LBNL Technical Report LBNL-40693, Lawrence Berkeley National Laboratory, 1997.
[20] D. Juric, G. Tryggvason, Computations of boiling flows, Int. J. Multiphase Flow 24 (3) (1998) 387–410.
[21] D. Martin, P. Colella, A cell-centered adaptive projection method for the incompressible Euler equations, J. Comput. Phys. 163 (2000).
[22] M.L. Minion, A projection method for locally refined grids, J. Comput. Phys. 127 (1996).
[23] J.Y. Poo, N. Ashgriz, A computational method for determining curvatures, J. Comput. Phys. 84 (1989) 483–491.
[24] E.G. Puckett, A.S. Almgren, J.B. Bell, D.L. Marcus, W.G. Rider, A high-order projection method for tracking fluid interfaces in variable density incompressible flows, J. Comput. Phys. 130 (1997) 269–282.
[25] Y. Renardy, M. Renardy, Prost: a parabolic reconstruction of surface tension for the volume-of-fluid method, J. Comput. Phys. 183 (2002) 400–421.
[26] W.J. Rider, D.B. Kothe, S. Jay Mosso, J.H. Cerutti, J.I. Hochstein, Accurate solution algorithms for incompressible multiphase flows, AIAA paper 95-0699, October 30, 1994.
[27] C.-W. Shu, Total-variation-diminishing time discretizations, SIAM J. Sci. Stat. Comput. 9 (6) (1988) 1073–1084.
[28] G. Son, V.K. Dhir, Numerical simulation of film boiling near critical pressures with a level set method, J. Heat Transfer 120 (1998) 183–192.
[29] M. Sussman, A. Almgren, J. Bell, P. Colella, L. Howell, M. Welcome, An adaptive level set approach for incompressible two-phase flows, J. Comput. Phys. 148 (1999) 81–124.
[30] M. Sussman, E.G. Puckett, A coupled level set and volume of fluid method for computing 3d and axisymmetric incompressible two-phase flows, J. Comput. Phys. 162 (2000) 301–337.
[31] M. Sussman, P. Smereka, Axisymmetric free boundary problems, J. Fluid Mech. 341 (1997) 269–294.
[32] M. Sussman, P. Smereka, S.J. Osher, A level set approach for computing solutions to incompressible two-phase flow, J. Comput. Phys. 114 (1994) 146–159.
[33] W.G. Szymczak, J. Rogers, J. Solomon, A.E. Berger, A numerical algorithm for hydrodynamic free boundary problems, J. Comput. Phys. 106 (1993) 319–336.
[34] B. van Leer, Towards the ultimate conservative difference scheme. v. a second-order sequel to godunov's method, J. Comput. Phys. 32 (1979) 101–136.
[35] M. Williams, D. Kothe, E.G. Puckett, Convergence and accuracy of kernel-based continuum surface tension models, in: Proceedings of the Thirteenth US National Congress of Applied Mechanics, Gainesville, FL, June 16–21, 1998.
[36] Y.L. Zhang, K. Yeo, B.C. Khoo, C. Wang, 3d jet impact and toroidal bubbles, J. Comput. Phys. 166 (2001) 336–360.