A STABLE AND EFFICIENT METHOD FOR TREATING SURFACE TENSION IN INCOMPRESSIBLE TWO-PHASE FLOW [*]

M. SUSSMAN[†] AND M. OHTA[‡]

This paper is dedicated to Diane Ohlhaber.

Abstract. A new approach based on volume preserving motion by mean curvature for treating surface tension in two-phase flows is introduced. Many numerical tests and a theoretical justification are included which provide evidence regarding the efficacy of the new approach. For many flows, which exhibit stiff surface tension effects, the new approach gives a factor of at least three and sometimes five or more speed-up for a given accuracy. The new method is easy to implement in the context of (1) level set methods, or coupled level set and volume-of-fluid methods, (2) complicated interfaces separating gas from liquid, and (3) three dimensional axisymmetric, or fully three dimensional adaptive mesh refinement.

Key words. surface tension, level set method, volume-of-fluid method, two-phase flow, motion by mean curvature, numerical methods

AMS subject classifications. 65M06, 76D05, 76T05

1. Introduction.

Explicit treatments of surface tension have a stringent stability constraint on the time step given by,

$$\Delta t \le \sqrt{\frac{\rho_L + \rho_G}{(2\pi)^3 \sigma}} \Delta x^{3/2}, \tag{1}$$

where ρ_L is the liquid density, ρ_G is the gas density, Δx is the size of a grid cell, and σ is the surface tension coefficient. In this paper, we shall introduce a new method for treating surface tension, based on volume preserving motion by mean curvature, which replaces the above time step constraint with the following, much more lenient, time step constraint:

$$\Delta t \le \frac{\Delta x(\rho_L + \rho_G)}{2\pi}. \tag{2}$$

Observe that our new constraint does not depend on the surface tension coefficient σ and our new constraint is restricted by $O(\Delta x)$ instead of $O(\Delta x^{3/2})$. If one further considers the CFL constraint,

$$U \Delta t < \Delta x, \tag{3}$$

then one can appropriately scale the Navier-Stokes equations so that U and $\frac{2\pi}{\rho_L + \rho_G}$ have comparable magnitudes. In this case, there is effectively no time step constraint due to surface tension. In other words, given any surface tension coefficient σ, one can always find an appropriate scaling of the Navier-Stokes equations so that our treatment for surface tension is unconditionally stable.

There have been efforts in the past to remove this surface tension time step constraint. For example Hou et al.[7] introduced a treatment for removing the stiff time

[*] Work supported in part by the National Science Foundation under contract DMS 0713256

[†] Department of Mathematics, Florida State University, Tallahassee, FL (sussman@math.fsu.edu). Questions, comments, or corrections to this document may be directed to that email address.

[‡] Department of Applied Chemistry, Muroran Institute of Technology, Muroran, Hokkaido, Japan

step constraint in the context of the boundary integral method for inviscid flows and Slikkerveer et al.[13] introduced an implicit surface tension treatment in which body fitted grids were used. A generalization of these approaches to problems with complex interfaces is not straightforward. Hochstein and Williams[6] derived an expression for the predicted time advanced curvature in order to remove the stiffness from the surface tension term; but their approach is not complete as in order to truly predict the time advanced curvature in an implicit fashion, one must also take into account the time advanced pressure gradient, which these authors did not do. A similar oversight was made by Cohen and Molemaker[1] who introduced a temporal sub-cycling procedure in which the stringent time step constraint was only applied to the surface tension force term. As with [6], Cohen and Molemaker only applied their treatment to the surface tension force term, while the pressure gradient term used the larger time step. While the methods proposed by [6] and [1] might work on some simple test problems, there is little numerical evidence or theoretical justification that these methods would work in a general context. Finally, an implicit level set treatment for surface tension was introduced by Hysing[8] and used by Raessi et al.[11], but only very simple test cases in two dimensions were attempted in which the interface did not merge or break up. The robustness of Hysing's approach to three dimensions, adaptive mesh refinement, or complex interfaces is unknown.

What distinguishes our approach from previous approaches is that we do not attempt to derive some kind of implicit or semi-implicit formulation for surface tension. Such a formulation is inevitably complicated since a coupling relation must be derived between velocity and the location of the interface. Instead, we show that one can simply replace $\sigma\kappa$, where σ is the surface tension coefficient, and κ is the curvature, by the quantity,

$$\frac{d(\Delta t) - d(0)}{\Delta t}$$

where $d(\tau)$ is the solution for volume preserving motion by mean curvature,

$$d_\tau = \sigma(\kappa - \kappa_{avg})|\nabla d|. \tag{4}$$

So, instead of trying to derive efficient implicit/semi-implicit surface tension techniques, we have reduced the problem of stiff surface tension to the problem of trying to efficiently solve the much simpler problem of volume preserving motion by mean curvature (4).

2. Method for removing the stiff surface tension time step constraint . The surface tension force term appears in the Navier Stokes equations for two phase flow:

$$\rho\frac{D\boldsymbol{U}}{Dt} = \nabla\cdot(-pI + 2\mu D) + \rho g\hat{\boldsymbol{z}} - \sigma\kappa\nabla H \tag{1}$$

$$\nabla\cdot\boldsymbol{U} = 0$$

$$\frac{D\phi}{Dt} = 0 \tag{2}$$

$$\frac{DF}{Dt} = 0 \tag{3}$$

$$\rho = \rho_L H(\phi) + \rho_G(1 - H(\phi))$$

$$\mu = \mu_L H(\phi) + \mu_G(1 - H(\phi))$$

$$\kappa(\phi) = \nabla \cdot \frac{\nabla\phi}{|\nabla\phi|} \tag{4}$$

$$H(\phi) = \begin{cases} 1 & \phi \geq 0 \\ 0 & \phi < 0 \end{cases} \tag{5}$$

The force term due to surface tension is given by,

$$-\frac{\sigma\kappa(\phi)\nabla H(\phi)}{\rho(\phi)}, \tag{6}$$

where $\kappa(\phi)$ is the curvature, $H(\phi)$ is the Heaviside function (5), ϕ is a level set function which is positive in the liquid and negative in the gas, F is a volume-of-fluid function which represents the volume fraction of liquid in each computational cell, σ is the surface tension coefficient, and $\rho(\phi)$ is the density.

Our new numerical method for treating (6) is the same as that proposed by the "ghost fluid" treatments of Kang et al[9] or Sussman et al[19] except that we treat the term $\sigma\kappa(\phi)$ differently. In our new approach, we determine $\sigma\kappa(\phi)$ by solving the volume preserving motion by mean curvature problem,

$$d_\tau = \sigma(\kappa(d) - \kappa_{avg}(d)), \qquad d(\boldsymbol{x}, 0) = \phi(\boldsymbol{x}) \tag{7}$$

for fictitious time $\tau = 0, \ldots, \Delta t$ and then taking

$$\sigma\kappa(\phi) \equiv \frac{d(\boldsymbol{x}, \Delta t) - d(\boldsymbol{x}, 0)}{\Delta t}.$$

Δt is the overall time step for integrating (1). In other words we implement the following algorithm for finding $\sigma\kappa(\phi)$:

1. define a temporary level set function $d(\boldsymbol{x}, 0) = \phi(\boldsymbol{x})$.

2. define $\Delta\tau$ so that $\Delta\tau < \frac{\Delta x^2}{4\sigma}$ (in two dimensions) and $N\Delta\tau = \Delta t$.

3. for $k = 0, \ldots, N - 1$,

4. find $\kappa(d^k)$ in computational cells (I, J) where $\phi_{I,J}\phi_{I',J'} \leq 0$ and $|\phi_{I,J}| \leq |\phi_{I',J'}|$. Cell (I', J') is within the 5 point "star" stencil (in 2d) about cell (I, J).

5. Determine a "color field" $C(d^k)$; C is zero away from the zero level set of d^k and where the level set function changes sign, C equals a unique "color" for each connected interfacial segment. For each unique color C, we determine the average curvature κ_{avg} and define $\tilde{\kappa} = \kappa(d^k) - \kappa_{avg}$. In other words, each connected interfacial segment has its own associated average curvature κ_{avg}.

6. Extend $\tilde{\kappa}$ in a narrow band about the zero level set of d^k. $\tilde{\kappa}$ approximately satisfies $\nabla\tilde{\kappa} \cdot \nabla d = 0$.

7. $d^{k+1} = d^k + \Delta\tau\sigma\tilde{\kappa}$.

8. extrapolate d^{k+1} in a narrow band about the zero level set of d^{k+1} without disturbing the zero level set of d^{k+1}.

9. endfor

10. Assign $\sigma\kappa(\phi)$ to be $\frac{d^N - d^0}{\Delta t}$.

Remarks:

- Since $\Delta\tau$ can be smaller than the original explicit surface tension time step constraint (1), it is important that the above steps be very efficient. In particular, step (5) above requires one to determine the color field $C(d^k)$ for a general "complicated" interface. We present an algorithm in section 5 that requires just a few sweeps of the grid, even for a block structured AMR grid on a parallel computer, in order to determine the color field $C(d^k)$.
- For the level set extension in step (8), we update the level set function d^{k+1} in such a way that we do not perturb the zero level set of d^{k+1} (i.e. we do not change d^{k+1} at cells in which d^{k+1} changes sign). When $k = N - 1$, the thickness of the extended narrow band must be large enough to contain the zero level set of d^0. See section 6 for the details of our level-set extension procedure.
- In previous work [15, 19, 17], we have computed curvature using the height function technique directly from the volume-of-fluid function F. In this paper, we introduce a height function technique for computing curvature from the level set function. We find our new approach, in terms of ease of implementation, and accuracy, to be superior to using the volume-of-fluid based height function approach. In section 4 we describe the details for how we discretize $\kappa(\phi)$ and in section 5.2 we describe the details for how we extrapolate $\kappa - \kappa_{avg}$ in a narrow band about the zero level set of ϕ.
- A theoretical justification, derived from linear stability analysis, for why our algorithm works is given in section 3. We remark, that our analysis also shows that one can replace our explicit (forward Euler) volume preserving motion by mean curvature algorithm (which uses a relatively small $\Delta\tau$) with an implicit (backward Euler) volume preserving motion by mean curvature algorithm (which can use a significantly larger $\Delta\tau$) and still preserve the same stability properties.

3. Theoretical justification for removing surface tension stiffness . We consider the linearized equations of (1) for deriving the time step stability constraint for our new algorithm. We assume zero viscosity and linearize about the trivial solution in which the velocity is identically zero, the free surface is located along the line $y = 0$, the gas region consists of the points $y > 0$ and the liquid region consists of the points $y < 0$. The linearized equations are:

$$\frac{d\boldsymbol{U}_G}{dt} = -\frac{\nabla p_G}{\rho_G} \qquad y > 0 \tag{1}$$

$$\nabla \cdot \boldsymbol{U}_G = 0 \qquad y > 0$$

$$\frac{d\boldsymbol{U}_L}{dt} = -\frac{\nabla p_L}{\rho_L} \qquad y < 0 \tag{2}$$

$$\nabla \cdot \boldsymbol{U}_L = 0 \qquad y < 0$$

$$\frac{1}{\rho_L}\frac{\partial p_L}{\partial y} = \frac{1}{\rho_G}\frac{\partial p_G}{\partial y} \qquad y = 0$$

$$p_L - p_G = -\sigma\kappa(\eta) \qquad y = 0 \tag{3}$$

$$\frac{d\eta}{dt} = v \qquad y = 0$$

$$\lim_{y\to\pm\infty} \boldsymbol{U} = 0$$

We assume the perturbed interfacial location is given by,

$$y = \eta(x,t) = f(t)e^{ikx},$$

the perturbed liquid pressure is

$$p_L(x,y,t) = \rho_L g(t)e^{ikx+ky},$$

and the perturbed gas pressure is

$$p_G(x,y,t) = -\rho_G g(t)e^{ikx-ky}.$$

The standard approach to treating surface tension would be to replace $\sigma\kappa(\eta)$ with $\sigma\eta_{xx} = -\sigma k^2\eta$. This results in the following equation for $f(t)$:

$$f'(t) = \pm iAf(t) \tag{4}$$

where

$$A = k^{3/2}\sqrt{\frac{\sigma}{\rho_L+\rho_G}}.$$

If one uses the forward Euler approach for solving (4) then the region of absolute stability requires that,

$$|\Delta t A| < 1. \tag{5}$$

In other words,

$$\Delta t \le \sqrt{\frac{\rho_L+\rho_G}{(2\pi)^3\sigma}}\Delta x^{3/2}, \tag{6}$$

where we have replaced k with $2\pi/\Delta x$. We remark, that if the ratio $\frac{\mu}{\sigma}$ is large (e.g. a fluid that is much more viscous than water), then a relaxed version of (6) has been derived by Galusinski and Vigneaux[2].

Now, suppose we compute $\sigma\kappa(\eta)$ (3) using the volume preserving motion by mean curvature algorithm described in section 2. In terms of our linearized variables, one has the following steps:

1. $d(x,0) = \eta(x,t)$

2. solve

$$d_\tau = \sigma d_{xx} \qquad \tau = 0,\ldots,\Delta t. \tag{7}$$

Since $\eta(x,t) = f(t)e^{ikx}$, one can explicitly write the solution for $d(x,\Delta t)$ in terms of η: $d(x,\Delta t) = e^{-k^2\sigma\Delta t}\eta(x,t)$.

3.

$$\sigma\kappa(\eta) = \frac{d(x,\Delta t) - d(x,0)}{\Delta t} = \frac{e^{-k^2\sigma\Delta t} - 1}{\Delta t}\eta(x,t) \tag{8}$$

In replacing $\sigma\kappa(\eta) = -k^2\sigma\eta$ with $\sigma\kappa(\eta) = \frac{e^{-k^2\sigma\Delta t}-1}{\Delta t}\eta$, our new expression for A in (4) becomes

$$A = \sqrt{k\frac{1 - e^{-k^2\sigma\Delta t}}{\Delta t(\rho_L + \rho_G)}}$$

and the time step constraint becomes

$$\Delta t \leq \frac{\Delta x(\rho_L + \rho_G)}{2\pi}.$$

Remarks:

- If we replace step 2 above (7) with the backwards Euler method, i.e.,

  $$\frac{d(x,\Delta t) - d(x,0)}{\Delta t} = \sigma d(x,\Delta t)_{xx},$$

  then

  $$\sigma\kappa(\eta) = \frac{d(x,\Delta t) - d(x,0)}{\Delta t} = -\frac{\sigma k^2}{1 + \sigma k^2\Delta t}\eta(x,t) \tag{9}$$

  and the resulting time step constraint becomes identical to (2). This indicates that one can accelerate the computation of surface tension driven flows if one had an efficient approach to solving volume preserving motion by mean curvature using the backward Euler time stepping scheme.
- Our proposed algorithm for evaluating the surface tension force is consistent with the equations for surface tension and introduces an $O(\Delta t)$ error to the numerical solution. Suppose one could solve (7) exactly and that $\tilde{\kappa} = \kappa(d) - \kappa_{avg}(d)$ is defined off of the zero level set of d in such a way so that $\nabla\tilde{\kappa}\cdot\nabla d = 0$ is always satisfied. In other words, $\tilde{\kappa}(\boldsymbol{x}_{ij},\tau)$ is the curvature of the interface at the point on the interface closest to the grid node $\boldsymbol{x}_{ij}$ at fictitious time τ. Also, since d will be a distance function, the curvature on the interface is defined as the laplacian of d. We integrate (7) with respect to τ in order to arrive at:

  $$d(\boldsymbol{x}_{ij},\Delta t) - d(\boldsymbol{x}_{ij},0) = \int_0^{\Delta t} \sigma\tilde{\kappa}(\boldsymbol{x}_{ij},\tau)d\tau. \tag{10}$$

  At smooth sections of the interface, we assume that the following relation holds,

  $$\tilde{\kappa}(\boldsymbol{x}_{ij},\tau) = \tilde{\kappa}(\boldsymbol{x}_{ij},0) + O(\tau). \tag{11}$$

  (11) is justified since the linearized version of (7) corresponds to the heat equation (7) and one can take the second derivative of both sides of (7) in order to show that the curvature for small values of τ satisfies a heat equation (7) too. Integrating the heat equation with respect ($\tilde{\kappa}_\tau = \sigma\tilde{\kappa}_{xx}$) to τ gives that $\tilde{\kappa}(x,\tau) = \tilde{\kappa}(x,0) + O(\tau)$. After substituting (11) into (10), one has

  $$\frac{d(\boldsymbol{x}_{ij},\Delta t) - d(\boldsymbol{x}_{ij},0)}{\Delta t} = \sigma\tilde{\kappa}(\boldsymbol{x}_{ij},0) + O(\Delta t).$$

- Relating to the previous remark, we have found in our test calculations that the $O(\Delta t)$ temporal error incurred by replacing $\sigma\kappa$ with $\frac{d(\boldsymbol{x},\Delta t)-d(\boldsymbol{x},0)}{\Delta t}$ is insignificant when compared to the spatial error. However, in the test problem on capillary instability (section 8.4), we have found that the temporal error associated with our new method can effect the overall accuracy. In these cases, in which the $O(\Delta t)$ error associated with our new method is noticeable, we propose that one could implement a "higher order stable method for treating surface tension." The $O(\Delta t^2)$ version of our algorithm would be to use a second order one-sided difference approximation for d_τ rather than the present first order one-sided difference approximation. The second order method is carried out by first integrating (7) for $\tau = 0, \ldots, 2\Delta t$ and then setting,

$$\sigma\kappa = \frac{4d(\boldsymbol{x},\Delta t) - 3d(\boldsymbol{x},0) - d(\boldsymbol{x},2\Delta t)}{2\Delta t}.$$

For the second order method, the expression for A that appears in the stability condition (5), is

$$A = \sqrt{k\frac{2(1-e^{-k^2\sigma\Delta t}) - (1/2)(1-e^{-2k^2\sigma\Delta t})}{\Delta t(\rho_L+\rho_G)}}$$

and the time step constraint becomes

$$\Delta t \le \frac{2}{3}\frac{\Delta x(\rho_L+\rho_G)}{2\pi}.$$

4. Level Set, Height function technique for computing curvature . In [17, 15, 19], the curvature was computed directly from the volume fractions using the "height function" technique. In this paper, we compute the curvature directly from the level set function.

For each cell (I, J), consider the 5 point stencil (in 2d) about this cell. If $\phi_{I,J}\phi_{I',J'} \le 0$ and $|\phi_{I,J}| \le |\phi_{I',J'}|$ where (I', J') is contained in the 5 point stencil of (I, J), then we compute the curvature as follows (please refer to Figure 4.1):

a. Determine the orientation of the interface based on $\nabla\phi_{I,J}$. Without loss of generality, assume the interface is oriented more horizontally than vertically ($|\phi_y| > |\phi_x|$).

b. Determine the heights h_{I-1}, h_I, h_{I+1} by looking at where the level set function changes sign for the columnar 1x7 data. e.g. to find h_{I-1}, one looks at the values $\phi_{I-1,K}$ where $K = J-3, \ldots, J+3$. One determines the sign change closest to $K = J$. If all the values of $\phi_{I-1,K}$ are negative and $\phi_{I-1,J+3} > \phi_{I-1,J-3}$, for example, then h_{I-1} takes the height of the highest point in the stencil.

c. Approximate the curvature as

$$\kappa_{I,J} = \frac{\frac{h_{I+1}-2h_I+h_{I-1}}{\Delta x^2}}{(1+(\frac{h_{I+1}-h_{I-1}}{2\Delta x})^2)^{3/2}}$$

Remarks:

- The coupled level-set and volume-of-fluid reinitialization step replaces the level set function with the signed distance to the piecewise linear volume-of-fluid reconstructed interface. The direct calculation of curvature from the reinitialized level set function will not converge under grid refinement as $\Delta x \to 0$. In section 8.1, we will show that the curvature *does* converge under grid refinement as $\Delta x \to 0$ and $\Delta t \to 0$ if we compute curvature of the level set function as a result of (1) reinitialization, and in addition, (2) applying volume preserving motion by mean curvature for $0 \le \tau \le \Delta t$.

- In light of the previous remark, it makes no difference whether one computes curvature directly from the volume fractions, or from the level set function. Being as such, computing the curvature from the level set function is preferable since it is (1) easier especially in the context of computing on an adaptive hierarchy of grids, and (2) more accurate in the sense that the position of the interface is well defined from the level set function especially when two interfaces come in close proximity to each other.
- From these authors' experience, the important thing to consider when discretizing the curvature for surface tension is that the curvature represent an approximation to the curvature *on the interface.* i.e. the height function approximation, whether one uses the level-set function ϕ or the volume-of-fluid function F is preferable to using the formula $\kappa(\phi) = \nabla \cdot \frac{\nabla\phi}{|\nabla\phi|}$ since the latter formula is an approximation to the curvature at the cell center, and the height function techniques approximate the curvature on the interface.

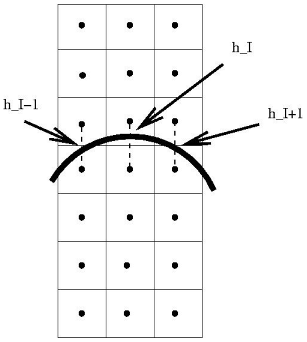

FIG. 4.1. *Illustration for calculating the curvature from the level set function. The heights,* h_{I-1}, h_I, h_{I+1} *are determined in between cells in which the level set function* ϕ *changes sign.*

5. Calculating the average curvature on a block structured adaptive hierarchy of grids .

The equation for volume preserving motion by mean curvature is

$$\phi_t = (\kappa(\phi) - \kappa_{avg}(\phi))$$

To compute κ_{avg}, we first define $\kappa(\phi)$ for cells (I, J) in which $\phi_{I,J}\phi_{I',J'} \le 0$ and $|\phi_{I,J}| \le |\phi_{I',J'}|$ ((I', J') is contained in the 5 point stencil of (I, J)) (see section 4). We define a variable $M_{I,J}$ which satisfies $M_{I,J} = 1$ where ever $\kappa_{I,J}$ is defined, and satisfies $M_{I,J} = 0$ elsewhere. Then, for a general complex interface, we determine a color field, $C(\phi)$ which is zero where ϕ does not change sign, and $C(\phi)$ equals a unique "color" C in cells where ϕ does change sign. For each color C, one has an associated κ_{avg} derived by the following discrete quadrature,

$$\kappa_{avg} = \frac{\sum_{i,j,C_{i,j}=C,M_{i,j}=1} \kappa_{ij} V_{i,j}}{\sum_{i,j,C_{i,j}=C,M_{i,j}=1} V_{i,j}}$$

where $V_{i,j}$ is the volume of cell (i, j).

5.1. Determining the interfacial color field . In this section we describe how we initialize a color field $C(d)$ where d is a level set function. In Figure 5.1, we illustrate an example of air/water interfaces along with the underlying adaptive hierarchy of grids. Each grid cell (I, J) in which $d_{I,J}d_{I',J'} \leq 0$ for some cell (I', J') contained in the 5 point stencil (2d) about (I, J) needs to be assigned a “color.” A unique color is defined for each “connected” interfacial piece. An air/water interface need not wholly be contained on the finest adaptive level.

The steps for assigning unique colors are as follows:

```
1. for each level $L$ from finest to coarsest,
2. for each grid $G$ on level $L$, assign local colors, do not include information from
      neighboring grids, and block out cells that lie underneath a finer level $L + 1$.
3. synchronize colors between grids on the same level $L$. A matrix $A$ is constructed
      in which $A_{I,J} = 1$ if there are two neighboring grid cells in which color $I$ is
      a neighbor of color $J$. Unique colors are derived, and the coloring scheme on
      level $L$ is updated.
4. Transfer color information from level $L + 1$ to level $L$ cells that were originally
      blocked out.
5. Synchronize colors between level $L$ and level $L+1$ and correct the color numbering
      scheme on all levels $L, \ldots, L_{finest}$.
6. end for.
```

In order to assign local colors for each grid G on level L (step 2 above), we follow these steps:

```
1. Initialize a local grid color counter $c = 0$. Initialize $C_{I,J} = 0$ for all cells $(I, J)$
      in grid $G$. Initialize $M_{I,J} = 1$ if cell $(I, J)$ is covered by a level $L + 1$ cell;
      otherwise $M_{I,J} = 0$.
2. For each cell $(I, J)$ of grid $G$ in which $M_{I,J} = 0$ and $C_{I,J} = 0$:
3. if $d_{I,J}d_{I',J'} \leq 0$ where $(I', J')$ is contained within the 5 point stencil of $(I, J)$ then:
      a. $c = c+1$, $C_{I,J} = c$, initialize a “stack” and push all the cells $(I', J')$ which
            are within the 5 point stencil of $(I, J)$, and satisfy $C_{I',J'} = 0$, $M_{I',J'} = 0$,
            onto the stack.
      b. while the stack is not empty,
      c. pop cell $(I'', J'')$ from the stack
      d. if $d_{I'',J''}d_{I''',J'''} \leq 0$ where $(I''', J''')$ is contained within the 5 point stencil
            of $(I'', J'')$, and $C_{I'',J''} = 0$, then:
            I. $C_{I'',J''} = c$, push all the cells $(I''', J''')$, which are within the 5 point
                  stencil of $(I'', J'')$, and have not been pushed before, and satisfy
                  $C_{I''',J'''} = 0$ and $M_{I''',J'''} = 0$, onto the stack.
      e. end while
4. endfor
```

5.2. Extrapolation algorithm for $\kappa - \kappa_{avg}$. After we determine $\tilde{\kappa} = \kappa - \kappa_{avg}$ in cells (I, J) where $\phi_{I,J}\phi_{I',J'} \leq 0$ and $|\phi_{I,J}| \leq |\phi_{I',J'}|$ ((I', J') is contained in the 5 point stencil of cell (I, J)), we tag these cells with the number “1” and do the following steps:

```
1. Traverse all cells $(i, j)$ in the domain, for each cell $(i, j)$ tagged with the number
      “1,” give a new tag (tag equal to the number “2”) to all cell within the 7x7
      stencil about $(i, j)$ that were not originally tagged.
2. Traverse all cells $(i, j)$ in the domain, for each newly tagged cell $(i, j)$ (tag equal
```

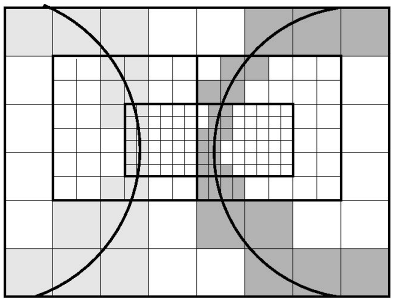

FIG. 5.1. *Illustration of the coloring of two separate interfacial segments on an adaptive hierarchy of grids.*

to “2”), assign an extrapolated curvature given by the formula,

$$\tilde{\kappa}_{i,j}^{extrap} = \frac{\sum_{i',j'=-3}^{3} w_{i',j'}\tilde{\kappa}_{i+i',j+j'}}{\sum_{i',j'=-3}^{3} w_{i',j'}}$$

$$(1)\; w_{i',j'} = \left\{ \begin{array}{cc} 0 & \text{cell } (i+i', j+j) \text{ is not originally tagged with tag “1”} \\ \frac{1}{(i'^2+j'^2)^5} & \text{cell } (i+i', j+j) \text{ is originally tagged with tag “1”} \end{array} \right.$$

6. Level set Extrapolation algorithm . Our volume preserving motion by mean curvature algorithm solves the following equation:

$$d_\tau = \sigma(\kappa(d) - \kappa_{avg}(d)) \qquad (1)$$

(1) is valid so long as d is maintained a distance function and $\nabla\tilde{\kappa} \cdot \nabla d = 0$. After each iteration of our volume preserving motion by mean curvature algorithm, we reinitialize d to be a distance function for cells within a narrow band of the zero level set of d, but without disturbing the position of the zero level set of d. We note that there are very efficient “fast marching” procedures [12, 3] for carrying out this step, but we have chosen the following very simple algorithm:

1. For each cell (I, J) in which $d_{I,J}d_{I',J'} \le 0$ for some (I', J') within the 9 point stencil about (I, J) (in 2d), tag this cell with the value “1.” All other cells are initially tagged with the value “0.”

2. For $M = 0, \ldots 2$,

3. For each cell (I, J) that is not tagged with “1,” but has at least one neighbor cell (within the 5 point stencil) that has a tag other than 0, solve the following equation for $d_{I,J}^{new}$,

$$(\frac{d_{I,J}^{new} - a}{\Delta x})^2 + (\frac{d_{I,J}^{new} - b}{\Delta y})^2 = 1$$

$$a = \min_{tag \neq 0}(|d_{I-1,J}|, |d_{I+1,J}|)$$

$$b = \min_{tag \neq 0}(|d_{I,J-1}|, |d_{I,J+1}|)$$

$$d_{I,J}^{new} = \text{sign}(d_{I,J})d_{I,J}^{new}$$

and set the new tag for cell (I, J) equal to “2.”

4. end for each (I, J)

5. end for M

Remark:

- Since $\sigma\kappa(\phi) = \frac{d^N - d^0}{\Delta t}$ where d^0 is the original level set function that is input into our volume preserving motion by mean curvature algorithm, and d^N is the output of our volume preserving motion by mean curvature algorithm, it is necessary that d^N be a distance function in a band wide enough that it contains the zero level set of d^0. So, at the very end of our motion by mean curvature algorithm, i.e. for the very last level set extrapolation step, it may be necessary to iterate for more than $M = 0, \ldots, 2$ iterations in order that d^N be a distance function that extends out beyond the zero level set of d^0.

7. Outline of overall algorithm for computational tests. Our numerical method is a "sharp interface" [9, 19, 14] method for multiphase flow. The projection method is used to enforce the divergence free condition for the velocity field. At a given time step n, we are given a face centered velocity field $\boldsymbol{u}^n_{face}$, cell centered velocity field $\boldsymbol{u}^n_{cell}$, cell centered level set function ϕ^n, and cell centered volume-of-fluid function F^n. An outline of our overall algorithm is as follows:

1. Advection (gas and liquid velocities treated separately):

$$\frac{\boldsymbol{u}^*_{cell} - \boldsymbol{u}^n_{cell}}{\Delta t} = [\boldsymbol{u} \cdot \nabla \boldsymbol{u}]^n$$

$$\frac{\phi^{n+1} - \phi^n}{\Delta t} = [\boldsymbol{u} \cdot \nabla \phi]^n$$

$$\frac{F^{n+1} - F^n}{\Delta t} = [\boldsymbol{u} \cdot \nabla F]^n$$

2. CLSVOF Reinitialization of the level set function; ϕ^{n+1} is replaced by the signed distance to the piecewise linear volume-of-fluid reconstructed interface (see Figure 7.1).

3. Diffusion:

a. define $\boldsymbol{u}^{(0)}_{cell} = \boldsymbol{u}^*_{cell}$.

b. define $\Delta\tau$ so that,

$$\Delta\tau < \min_{i,j} \frac{\rho_{i,j}\Delta x^2}{\mu_{i+1/2,j} + \mu_{i,j+1/2} + \mu_{i-1/2,j} + \mu_{i,j-1/2}} \qquad N\Delta\tau = \Delta t$$

c. for $k = 0, \ldots, N-1$,

$$\boldsymbol{u}^{(k+1)}_{cell} = \boldsymbol{u}^{(k)}_{cell} + \Delta\tau \frac{1}{\rho(\phi)} [\nabla \cdot \mu(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^T)]^{(k)}$$

4. extrapolate the cell centered advective force term, $F^{advect}_{cell} = \frac{\boldsymbol{u}^*_{cell} - \boldsymbol{u}^n_{cell}}{\Delta t}$, and diffusion force term, $F^{diffuse}_{cell} = \frac{\boldsymbol{u}^{(N)}_{cell} - \boldsymbol{u}^{(0)}_{cell}}{\Delta t}$ from cell centers to cell faces.

5. Construct face centered velocity field to be projected:

$$\boldsymbol{V}_{face} = \boldsymbol{u}^n_{face} + \Delta t F^{advect}_{face} + \Delta t F^{diffuse}_{face} - \Delta t [\sigma\kappa \frac{\nabla H}{\rho}]^{n+1} + \Delta t g \boldsymbol{z}$$

6. Pressure projection step:

$$\nabla \cdot \frac{\nabla p}{\rho} = \frac{1}{\Delta t} \nabla \cdot \boldsymbol{V}_{face}$$

$$\boldsymbol{u}^{n+1}_{face} = \boldsymbol{V}_{face} - \Delta t \frac{\nabla p}{\rho}$$

7. extrapolate the liquid velocity into the gas.

Remarks:

- The result of the nonlinear advection step $\boldsymbol{u}^{*}_{cell}$ is calculated in two steps:

  a. solve $\boldsymbol{u}_t + \boldsymbol{u}^n \cdot \nabla \boldsymbol{u} = 0$ using a 2nd order non-conservative semi-Lagrangian approach; i.e. find $\boldsymbol{u}^{*,predict}_{i,j,cell} = I\boldsymbol{u}^{n}_{cell}(\boldsymbol{x}^{*})$ where I represents cubic interpolation and $\boldsymbol{x}^{*}$ is the solution at $\tau = \Delta t$ of $\frac{d\boldsymbol{x}}{d\tau} = -\boldsymbol{u}^n(\boldsymbol{x})$ $(\boldsymbol{x}(t = 0) = \boldsymbol{x}_{ij})$.

  b. define

$$\boldsymbol{u}^{*,n+1/2}_{cell} = \frac{\boldsymbol{u}^{n}_{cell} + \boldsymbol{u}^{*,predict}_{cell}}{2}$$

$$\frac{u^{*}_{cell} - u^{n}_{cell}}{\Delta t} = \frac{u^{n}_{i+1/2,j} u^{*,n+1/2}_{i+1/2,j} - u^{n}_{i-1/2,j} u^{*,n+1/2}_{i-1/2,j}}{\Delta x} + \frac{v^{n}_{i,j+1/2} u^{*,n+1/2}_{i,j+1/2} - v^{n}_{i,j-1/2} u^{*,n+1/2}_{i,j-1/2}}{\Delta y}$$

$$\frac{v^{*}_{cell} - v^{n}_{cell}}{\Delta t} = \frac{u^{n}_{i+1/2,j} v^{*,n+1/2}_{i+1/2,j} - u^{n}_{i-1/2,j} v^{*,n+1/2}_{i-1/2,j}}{\Delta x} + \frac{v^{n}_{i,j+1/2} v^{*,n+1/2}_{i,j+1/2} - v^{n}_{i,j-1/2} v^{*,n+1/2}_{i,j-1/2}}{\Delta y}$$

- The first and second components of the surface tension term, $[\sigma\kappa \frac{\nabla H}{\rho}]^{n+1}$, are discretized as,

$$\frac{1}{\rho_L \theta_{i+1/2,j} + \rho_G (1-\theta)_{i+1/2,j}} \sigma\kappa_{i+1/2,j} \frac{H(\phi_{i+1,j}) - H(\phi_{i,j})}{\Delta x}$$

  and

$$\frac{1}{\rho_L \theta_{i,j+1/2} + \rho_G (1-\theta)_{i,j+1/2}} \sigma\kappa_{i,j+1/2} \frac{H(\phi_{i,j+1}) - H(\phi_{i,j})}{\Delta y}$$

  respectively. $H(\phi)$ is defined sharply as in (5). $\theta_{i+1/2,j}$ is a height fraction given by,

$$\theta_{i+1/2,j}(\phi) = \begin{cases} 1 & \phi_{i+1,j} \geq 0 \text{ and } \phi_{i,j} \geq 0 \\ 0 & \phi_{i+1,j} < 0 \text{ and } \phi_{i,j} < 0 \\ \frac{\phi^{+}_{i+1,j} + \phi^{+}_{i,j}}{|\phi_{i+1,j}| + |\phi_{i,j}|} & \text{otherwise} \end{cases}$$

  The "+" superscript stands for the "positive part:" i.e., $a^{+} \equiv \max(a, 0)$. The term $\sigma\kappa$ is determined at cell centers using our newly proposed volume preserving motion by mean curvature procedure as described in section 2. $\sigma\kappa$ is extrapolated to cell faces as follows:

$$\sigma\kappa_{i+1/2,j} = \begin{cases} \sigma\kappa_{i,j} & |\phi_{i,j}| < |\phi_{i+1,j}| \\ \sigma\kappa_{i+1,j} & |\phi_{i,j}| \geq |\phi_{i+1,j}| \end{cases}$$

- Referring to the surface tension stability constraint (2), As long as the Navier-Stokes equations are scaled appropriately so that the dimensionless velocity has comparable size as the quantity $\frac{2\pi}{\rho_L + \rho_G}$, then the only stability constraint for our algorithm is the CFL condition: $\max U \Delta t < \Delta x$.

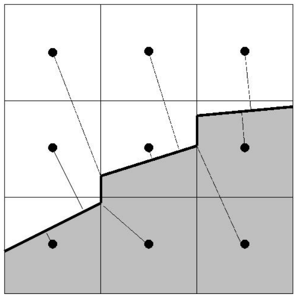

Fig. 7.1. *The CLSVOF reinitialization step replaces the original level set function ϕ with the signed distance from the piecewise linear volume-of-fluid reconstructed interface. The volume fraction of a cell that is shaded corresponds to the volume-of-fluid function F and the slope of the reconstructed interface corresponds to $\frac{\nabla\phi}{|\nabla\phi|}$. The level set function is positive in shaded regions and negative outside of shaded regions.*

Table 8.1

Convergence study for curvature when computed from the level set function after reinitialization and volume preserving motion by mean curvature for $0 \le \tau \le \Delta t$. The curvature is measured for a circle with radius 1/2. Expected curvature is 4. (3d axisymmetric coordinate system)

Δx	Δt	minimum curvature	maximum curvature
1/32	0.004	3.981	4.010
1/64	0.002	3.993	4.003
1/128	0.001	3.996	4.001

8. Results.

8.1. Numerical validation of computing curvature from the level set function instead of the volume-of-fluid function .

In this section we test our level set "height function" approach for computing curvature (see section 4). For our first test, our domain is a 1x2 box and we have a 3d axi-symmetric coordinate system. We initialize a circle of radius 1/2 at $(r, z) = (0, 1)$. I.e., we initialize the level set function to be $\phi(r, z) = \sqrt{r^2 - (z-1)^2} - 1/2$ and the volume-of-fluid function F to represent the fraction of volume occupied by liquid in each computational cell. We then do a grid refinement study for the combination of steps in which we first reinitialize ϕ by replacing ϕ with the signed normal distance to the volume-of-fluid reconstructed interface, and then we modify ϕ via our volume preserving motion by mean curvature algorithm for $\tau = 0, \ldots, \Delta t$. The results of our first test are in table 8.1 where we measure at least first order convergence for the curvature.

Our second test is the same as the first test except that our domain is a 1x1x1 box and we have a fully 3d coordinate system. We initialize a circle of radius 1/2 at $(x, y, z) = (0, 0, 0)$. The results of our second test are reported in table 8.2.

8.2. Parasitic Currents .

In this section we test our implementation of surface tension for the problem of a static two-dimensional (2d) drop with unit diameter $D = 1$. We assume that the surface tension coefficient is $\sigma = 1.2$, the liquid viscosity is $\mu_L = 0.01$, the liquid density is $\rho_L = 1$, and the density ratio and viscosity ratio are both one. The exact solution for such a problem is that the velocity $\boldsymbol{u}$ is identically zero. We remark that our choice of parameters correspond to an Ohnesorge number

Table 8.2

Convergence study for curvature when computed from the level set function after reinitialization and volume preserving motion by mean curvature for $0 \leq \tau \leq \Delta t$. The curvature is measured for a circle with radius 1/2. Expected curvature is 4. (fully 3d coordinate system)

Δx	Δt	minimum curvature	maximum curvature
1/32	0.004	3.979	4.007
1/64	0.002	3.993	4.002
1/128	0.001	3.998	4.001

Oh satisfying $1/Oh^2 = 12000$. The Ohnesorge number Oh is defined as,

$$Oh = \frac{\mu}{\sqrt{\sigma \rho D}}.$$

If we were to scale the Navier-Stokes equations by the time scale $T = D\mu/\sigma = 1/120$, and by the velocity scale $U = \sigma/\mu = 120$, then the non-dimensionalized Navier-Stokes equations become,

$$\frac{D\boldsymbol{u}}{Dt} = -\nabla p + Oh^2 \Delta \boldsymbol{u} - Oh^2 \kappa \nabla H. \tag{1}$$

We remark that in the context of our stable surface tension algorithm, it would be a very bad strategy to scale the equations according to (1). This is because it is important that the equations be scaled in such a way so that the non-dimensionalized velocity have magnitudes greater than or equal to $\frac{2\pi}{\rho_L + \rho_G}$. Recall that for our stable surface tension algorithm, the time step constraint associated with surface tension is given by (2):

$$\Delta t \leq \frac{\Delta x(\rho_L + \rho_G)}{2\pi}.$$

It is important that this condition be less stringent than the CFL constraint,

$$U \Delta t < \Delta x. \tag{2}$$

So, for our tests, we use the unscaled parameters for surface tension ($\sigma = 1.2$) and viscosity ($\mu = 0.01$) instead of scaling velocity by a factor of 120.

The dimensions of our computational grid are 5/2x5/2 with periodic boundary conditions at the left and right boundaries and reflecting boundary conditions at the top and bottom boundaries. A drop with unit diameter is initially located at the center of our domain $(5/4, 5/4)$. Our tolerance for the pressure solver is $1.0E-12$ (the error is measured as an absolute error and is the L^2 norm of the residual).

Our first test validates our height function algorithm for computing curvatures from the level set function ϕ, instead of from the volume-of-fluid function F. We do not incorporate our volume preserving motion by mean curvature algorithm into this first comparison. In Tables 8.3 and 8.4 we display the maximum velocity magnitude for varying grid resolutions at the time $t = 250/120$. The first table shows results using F to compute curvature, and the second table shows results using ϕ. These results are comparable with our previous results [19] (taking into account that the previous results had been scaled the velocity by 120).

Our second test validates our stable surface tension algorithm. In Table 8.5 we display the maximum velocity magnitude for varying grid resolutions at the time

Table 8.3

Convergence study for static droplet with surface tension (parasitic currents test). Curvature computed using volume-of-fluid height function technique. Stable surface tension algorithm not implemented. Maximum velocity at $t = 250/120$ is shown. $Oh^2 = 1/12000$.

Δx	Δt	maximum velocity
2.5/16	5.6E-3	1.2E-3
2.5/32	1.9E-3	3.1E-4
2.5/64	7.0E-4	1.5E-5

Table 8.4

Convergence study for static droplet with surface tension (parasitic currents test). Curvature computed using level set height function technique. Stable surface tension algorithm is not implemented. Maximum velocity at $t = 250/120$ is shown. $Oh^2 = 1/12000$.

Δx	Δt	maximum velocity
2.5/16	5.6E-3	4.9E-3
2.5/32	1.9E-3	1.2E-4
2.5/64	7.0E-4	1.6E-6

$t = 250/120$. These results demonstrate that our new algorithm enables comparable accuracy with a factor of 6 improved computation speed. For the case when $\Delta x = 2.5/64$, it took 44 seconds to run to completion when using our new stable surface tension approach, whereas it took 253 seconds without our new approach.

8.3. Surface tension driven (zero gravity) drop oscillations . In this section we compare our new method for treating surface tension to the method proposed in [19] for the problem of surface tension driven drop oscillations. In section 8.2 (parasitic currents), the density ratio was 1:1 and the viscosity ratio was 1:1; in this example, the density ratio is 1000:1 and the viscosity ratio is 1000:1.

According to the linearized results derived by Lamb [10] (1932, §275), the position of the drop interface is

$$R(\theta, t) = a + \epsilon P_n(\cos(\theta)) \sin(\omega_n t + \pi/2),$$

where

$$\omega_n^2 = \sigma \frac{n(n-1)(n+1)(n+2)}{a^3(\rho_l(n+1) + \rho_g n)}$$

and P_n is the Legendre polynomial of order n. θ runs between 0 and 2π, where $\theta = 0$ corresponds to $r = 0$ and $z = a$. If viscosity is present, Lamb (1932, §355) found that the amplitude is proportional to $e^{-t/\tau}$, where

$$\tau = \frac{a^2 \rho_L}{\mu_L(2n+1)(n-1)}.$$

We compute the evolution of a drop with $a = 1$, $g = 0$, $\mu_L = 1/50$, $\mu_L/\mu_G = 1000$, $\sigma = 1/2$, $\rho_L = 1$ and $\rho_L/\rho_G = 1000$. The initial interface is given by $R(\theta, 0)$, with $\epsilon = .05$ and $n = 2$. With these parameters we find $\omega_2 = 2.0$ and $\tau = 5.0$. The fluid domain is $\Omega = \{(r, z) | 0 \le r \le 1.5 \text{ and } 0 \le z \le 1.5\}$. Symmetric boundary conditions are imposed at $r = 0$ and $z = 0$.

Table 8.5

Convergence study for static droplet with surface tension (parasitic currents test). Curvature computed using level set height function technique. Stable surface tension algorithm is implemented. Maximum velocity at $t = 250/120$ *is shown.* $Oh^2 = 1/12000$.

Δx	Δt	maximum velocity
2.5/16	0.025	5.9E-4
2.5/32	0.012	9.2E-5
2.5/64	0.006	1.8E-6

In Table 8.6, we display the relative error between succeeding resolutions for the minor amplitude $R_{\Delta x}(0,t)$ of the droplet. The average error $E^{avg}_{Amplitude}$ is given by

$$E^{avg}_{amplitude} \equiv \int_0^{3.5} |R_{\Delta x}(0,t) - R_{2\Delta x}(0,t)|dt,$$

and the maximum amplitude error $E^{max}_{Amplitude}$ is given by

$$E^{max}_{Amplitude} \equiv \max_{0\le t\le 3.5} |R_{\Delta x}(0,t) - R_{2\Delta x}(0,t)|.$$

In Figure 8.1, we plot the minor amplitude computed on an 128x128 grid when (A) utilizing our stable surface tension algorithm with $\Delta t = 0.0019$, (B) disabling our stable surface tension algorithm, but still computing curvature from the level set function $\Delta t = 0.00018$, (C) disabling our stable surface tension algorithm, and computing curvature directly from the volume-of-fluid function $\Delta t = 0.00018$, and (D) utilizing our stable surface tension algorithm with $\Delta t = 0.0019$ and also utilizing adaptive mesh refinement with an effective fine grid resolution equivalent to 128x128 (see Figure 8.2 for an illustration of the adaptive grids for this problem). As can be seen from Figure 8.1, there is very little difference in the results whether we use the volume-of-fluid function or the level set function in calculating the curvature, and there is very little difference in the results whether we use our stable surface tension algorithm, or if we disable our stable surface tension algorithm. The big difference is that our new algorithm is much faster. In Table 8.7, we compare the run times with/without the implementation of our new stable surface tension algorithm.

Remarks:

- When we disable our stable surface tension algorithm, but force $\Delta t = 0.0019$ instead of $\Delta t = 0.00018$, the computations rapidly become unstable.
- In table 8.7 we indicate a *slower* computation time when using AMR (case E) versus not using AMR (case B). We speculate that this is because the blocking factor was 4 therefore restricting the multigrid preconditioner to only two levels when operating on the finest level.

8.4. Capillary Instability . In this section, we test our stable surface tension approach on the classical Rayleigh capillary instability problem in which a slightly perturbed cylindrical column of liquid is driven to break up into droplets by surface tension (capillary) effects. In this test problem we use physical properties that are comparable to those found in [18, 19].

We consider an initially perturbed cylindrical column of water in air. The shape of the initial interface is

$$r(z) = r_0 + \epsilon \cos(2\pi z/\lambda). \tag{3}$$

Table 8.6

Convergence study for zero gravity drop oscillations $\sigma = 1/2$, $\mu_L = 1/50$, $\mu_L/\mu_G = 1000$, $\rho_L/\rho_G = 1000$ and $\alpha = 2$. Curvature is computed using the level set function. Our stable surface tension algorithm is used to compute the surface tension force.

Δr	Δt	$E^{avg}_{Amplitude}$	$E^{max}_{amplitude}$
3/64	0.0075	N/A	N/A
3/128	0.0037	0.00174	0.00322
3/256	0.0019	0.00045	0.00114

Table 8.7

CPU time for zero gravity drop oscillations $\sigma = 1/2$, $\mu_L = 1/50$, $\mu_L/\mu_G = 1000$, $\rho_L/\rho_G = 1000$ and $\alpha = 2$. CPU times with/without our new stable surface tension algorithm are compared. The effective fine grid resolution is 128x128 in all cases. Case (A): new method, no AMR. Case (B): old method, no AMR. Case (C): old method, use VOF for curvature, no AMR. Case (D): new method, AMR. Case (E): old method, AMR.

case	Δt	cells per step	CPU time (seconds)
A	0.0019	16384	898
B	0.00018	16384	5379
C	0.00018	16384	5338
D	0.0019	6560	777
E	0.00018	6560	5590

We compute on a 3d-axisymmetric domain $\Omega = \{(r,z)|0 \leq r \leq \lambda/4 \text{ and } 0 \leq z \leq \lambda/2\}$. Symmetric boundary conditions are enforced at $r = 0$, $z = 0$ and $z = \lambda/2$. Outflow (pressure equals zero) boundary conditions are enforced at $r = \lambda/4$. The relevant dimensional parameters for this test problem are $r_0 = 6.52$ microns, $\epsilon = 1.3$ microns, $\lambda = 60$ microns, $\mu_L = 1.138 \times 10^{-2} g/(cm \cdot s)$, $\mu_G = 1.77 \times 10^{-4} g/(cm \cdot s)$, $\rho_L = 1.0 g/cm^3$, $\rho_G = 0.001225 g/cm^3$, and $\sigma = 72.8 dynes/cm$. In our computations we use the following dimensionless parameters: the Reynolds number $R = \rho_L LU/\mu_L = 1/1.138$, the Weber number $W = \rho_L LU^2/\sigma = 1.0/72.8$, $L = 1.0$ microns, $U = 1.0 m/s$ and the density and viscosity ratios are 816 and 64, respectively.

In Figures 8.3 and 8.4, we display the results of adaptive computations for the capillary jet using our stable surface tension approach when $\Delta t = 0.01$ and $\Delta t = 0.004$ respectively. In Figures 8.5 and 8.6 we show results with our stable surface tension algorithm disabled and the curvature is computed from the level set function and the volume-of-fluid function respectively. The time step for these cases were $\Delta t = 0.0013$. In Figure 8.7, we show the minimum jet radius versus time for the four cases: (a) $\Delta t = 0.01$, (b) $\Delta t = 0.004$, (c) $\Delta t = 0.0013$ using ϕ for curvature, and (d) $\Delta t = 0.0013$ using F for curvature.

The computation time for the adaptive $\Delta t = 0.01$ case was 1016 seconds and for the adaptive $\Delta t = 0.004$ case was 1435 seconds. On average, 4608 cells were advanced per time step for the adaptive cases. For the non-adaptive $\Delta t = 0.01$ case, the computation time was 982 seconds. For the non-adaptive $\Delta t = 0.0013$ case, the computation time was 2146 seconds.

Remarks:

- if we force the time step to be $\Delta t = 0.01$ and disable our stable surface tension algorithm, then the 64x128 computation quickly becomes unstable.
- Here we observe a speed-up when using our stable surface tension algorithm, but the results for $\Delta t = 0.01$ are not equivalent to the $\Delta t = 0.0013$ case; the

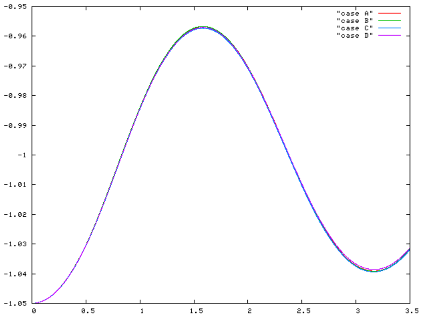

Fig. 8.1. *Minor amplitude for zero gravity drop oscillations. Effective fine grid resolution is 128x128. Case A: Stable algorithm enabled, level set function used to calculate curvature, no AMR. Case B: Stable algorithm disabled, level set function used to calculate curvature, no AMR. Case C: Stable algorithm disabled, volume-of-fluid function used to calculate curvature, no AMR. Case D: Stable algorithm enabled, level set function used to calculate curvature, AMR.*

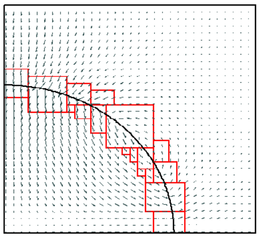

Fig. 8.2. *Snapshot at $t = 1.13$ of droplet vibrating due to surface tension induced oscillations. Effective fine grid resolutions is 128x128. One level of adaptivity overlays a base coarse level.*

pinch-off times are different. For this test case, surface tension is the *only* force causing break-up of the liquid jet, and therefore it is important to take small time steps in order to resolve the break-up process.

- Here we observed that the adaptive calculation of this problem took longer than the non-adaptive calculation. We speculate that this is because the blocking factor was 8 therefore restricting the multigrid preconditioner to only three levels when operating on the finest level.

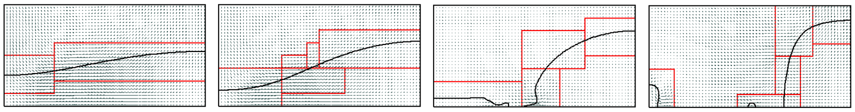

Fig. 8.3. *Results for the break-up of a liquid jet due to surface tension. Our stable surface tension algorithm is enabled and $\Delta t = 0.01$. The effective fine grid resolution is 64x128. One level of adaptivity overlays a 32x64 coarse level. The dimensionless times for each snapshot, when viewed from left to right, top to bottom are $t = 4.69$, $t = 9.38$, $t = 11.72$ and $t = 12.90$.*

8.5. Bubble Dynamics. In this section, we compute the steady state shape of a gas bubble rising in a viscous Newtonian liquid. For comparison, we use the experimental results found in [5] and computational results in [19].

We test our new stable surface tension algorithm on the bubble problem in Hnat and Buckmaster [5] corresponding to diameter $D = 1.215cm$, $\rho = 0.876g/cm^3$, $\mu =$

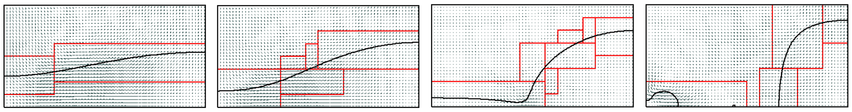

Fig. 8.4. *Results for the break-up of a liquid jet due to surface tension. Our stable surface tension algorithm is enabled and* $\Delta t = 0.004$*. The effective fine grid resolution is 64x128. One level of adaptivity overlays a base coarse level. The dimensionless times for each snapshot, when viewed from left to right, top to bottom are* $t = 4.69$*,* $t = 9.38$*,* $t = 11.72$ *and* $t = 12.90$.

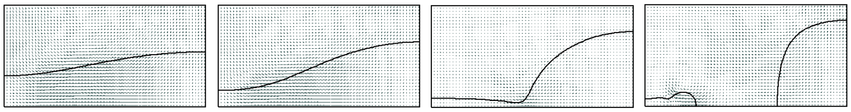

Fig. 8.5. *Results for the break-up of a liquid jet due to surface tension. Our stable surface tension algorithm is disabled and* $\Delta t = 0.0013$*. The grid resolution is 64x128. The curvature is computed using the level set height function technique. The dimensionless times for each snapshot, when viewed from left to right, top to bottom are* $t = 4.69$*,* $t = 9.38$*,* $t = 11.72$ *and* $t = 12.90$.

$1.18P$, and $\sigma = 32.2 dyne/cm$. The density ratio and viscosity ratio are 694:1 and 6557:1 respectively.

If one were to scale the Navier-Stokes equations by the diameter of the bubble, $D = 1.215cm$, and by the experimentally observed terminal rise speed of the bubble, $U = 21.5cm/s$, then the Reynolds number is, $R = \frac{\rho DU}{\mu} = 19.4$, the Weber number is, $We = \frac{\rho DU^2}{\sigma} = 15.30$ and the Froude number is $Fr = \frac{U^2}{gD} = 0.388$. This scaling would be an inefficient scaling when used together with our stable surface tension algorithm since the CFL stability condition is determined by the scaled rise speed of the bubble $U_{dimensionless} = 1$ which is less than the stable surface tension stability condition determined by the fictitious speed of $U_{tension} = \frac{2\pi}{\rho_L + \rho_G}$ (2). So, for this test, instead of scaling velocity by $U = 21.5cm/s$, we instead scale by a velocity of $U = 2.15cm/s$. Now, the expected rise speed of the bubble shall be $U_{dimensionless} = 10$ which dominates $U_{tension}$ which effectively eliminates our need for a surface tension time step constraint for this problem. In other words, the time step constraint is completely dictated by the velocity of the flow field instead of by surface tension effects. In terms of our new scaling, $R = 1.94$, $We = 0.1530$, and $Fr = 0.00388$.

We performed four tests, two tests using a 3d axi-symmetric coordinate system, and two tests using a fully 3d coordinate system. For our 3d axi-symmetric tests, the computational domain size was $0 \le r \le 4$ and $0 \le z \le 6$. For our fully 3d tests, the computational domain size was $0 \le x \le 4$, $0 \le y \le 4$, and $0 \le z \le 6$. The 3d tests assumed symmetry boundary conditions at $x = 0$ and $y = 0$. For the 3d axisymmetric tests, we initialized a spherical bubble at $(r, z) = (0, 3)$ with radius 1/2. For the 3d tests, we initialized a spherical bubble at $(x, y, z) = (0, 0, 3)$ with radius 1/2. We imposed inflow boundary conditions at the top of our computational domain corresponding to the expected rise speed of the bubble (expected dimensionless rise speed is 10). We imposed hydrostatic pressure boundary conditions at all other walls (except for the symmetry walls).

The computational grid is constructed based on block structured adaptive mesh refinement in which one has a hierarchy of successively refined overlapping levels. For our 3d axisymmetric calculations, the base coarse grid has dimensions of 48x72 and we have 3 added levels of adaptivity. For our fully 3d calculations, the base coarse

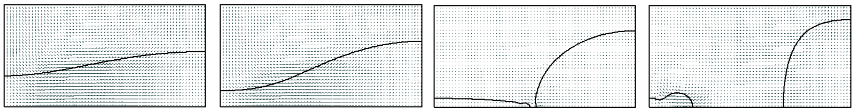

Fig. 8.6. *Results for the break-up of a liquid jet due to surface tension. Our stable surface tension algorithm is disabled and* $\Delta t = 0.0013$*. The grid resolution is 64x128. The curvature is computed using the volume-of-fluid height function technique. The dimensionless times for each snapshot, when viewed from left to right, top to bottom are* $t = 4.69$*,* $t = 9.38$*,* $t = 11.72$ *and* $t = 12.90$.

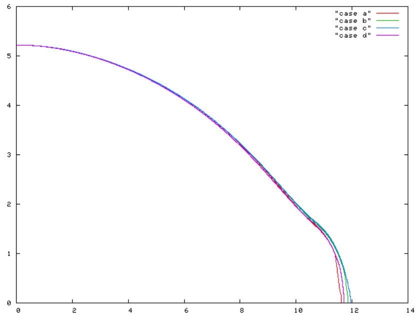

Fig. 8.7. *Minimum jet radius versus time for four cases: (a)* $\Delta t = 0.01$*, (b)* $\Delta t = 0.004$*, (c)* $\Delta t = 0.0013$ *using* ϕ *for curvature, and (d)* $\Delta t = 0.0013$ *using* F *for curvature.*

grid has dimensions of 32x32x48 and we have 2 added levels of adaptivity.

In all of our calculations we used the level set height function technique (see section 4) for computing the curvature instead of the volume-of-fluid height function technique as proposed in [19]. We compared results with/without our stable surface tension algorithm. A comparison of computed terminal bubble rise velocity versus previous computational and experimental results are reported in Table 8.8. A comparison of computed terminal bubble shapes with/without our stable surface tension algorithm are reported in Figures 8.8 and 8.10 corresponding to 3d axisymmetric and 3d results respectively. A comparison of the computed terminal bubble shape using our stable surface tension algorithm versus experimental results [5] is shown in Figure 8.9. As can be seen from our results, there is very little difference in terminal bubble shape or terminal bubble velocity whether one uses our stable surface tension approach or not. If one were to compare our results with those reported in [19], there is very little difference between calculating the curvature using our level set height function technique vs. the volume-of-fluid height function technique.

What we have found from this test is that it takes a fraction of the time to compute the same results using our stable surface tension algorithm. For the 3d axisymmetric test case, our stable surface tension approach took 3120 seconds to advance to dimensionless time $T = 0.84$ ($\Delta t = 4.8E-4$) while it took 9678 seconds to advance to the same time with our stable surface tension algorithm disabled ($\Delta t = 4.14E-5$). For the fully 3d test case, our stable surface tension approach took 312 minutes to advance to dimensionless time $T = 1.03$ ($\Delta t = 1.4E-3$) while it took 1016 minutes to advance to the same time with our stable surface tension algorithm disabled ($\Delta t = 2.2E-4$). In both the 3d axisymmetric and fully 3d cases, if we had disabled our stable surface tension algorithm, but retained the stable surface tension time step, then the calculation rapidly becomes unstable.

Table 8.8
Comparison of computed terminal bubble rise speed (units cm/s) compared with experiments (Hnat and Buckmaster) and compared with previous calculations (Sussman et al).

case	terminal rise speed (cm/s)
Hnat and Buckmaster (D=12.15mm)	21.5
Sussman et al (3d r-z)	21.8
Sussman et al (3d)	21.6
Present method w/Stable Surface Tension (3d r-z)	21.6
Present method w/Stable Surface Tension (3d)	21.6
Present method w/o Stable Surface Tension (3d r-z)	21.8
Present method w/o Stable Surface Tension (3d)	21.6

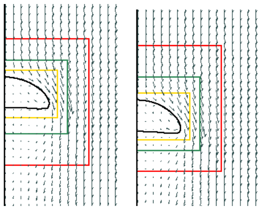

Fig. 8.8. *Terminal bubble shape for a gas bubble rising in a viscous liquid. The density ratio and viscosity ratio are 694:1 and 6557:1 respectively. Bubbles computed assuming a 3d axisymmetric coordinate system. The left result corresponds to $\Delta t = 4.8E-4$, $t = 9.5$, and the right result corresponds to $\Delta t = 4.14E-5$, $t = 0.83$.*

8.6. Bubble Formation. In this section we compute the formation of bubbles caused by the injection of air into a container of liquid. Our computations use 3d-axisymmetric r-z coordinates. We enforce inflow boundary conditions at the bottom of the domain ($z = 0$),

$$\nabla p \cdot \boldsymbol{n} = 0,$$

$$\boldsymbol{U} \cdot \boldsymbol{n} = \left\{ \begin{array}{cc} u_{inflow} & r < r_{nozzle} \\ 0 & \text{otherwise} \end{array} \right.$$

Symmetry boundary conditions are given at $r = 0$, free-slip conditions at $r = r_{high}$, and outflow conditions at the top of the domain ($z = z_{high}$):

$$p = 0.$$

Below we compare results using our stable surface tension algorithm with experimental results reported by Helsby and Tuson[4] and with computational results that we reported in[19]. Our target is Figure 1 (case-e) in [4]. This corresponds with a nozzle radius of $8.5E-4m$ and an inflow velocity of $0.44m/s$. Based on the physical properties of the case-e system, one has the Reynolds number equal to 3.6, the Weber number equal to 3.06, the density ratio equal to $1015 : 1$ and the viscosity ratio equal to $6923 : 1$. We used Adaptive Mesh Refinement[16, 14] to compute the solutions for the bubble formation problem with a base coarse grid of 32x96 grid cells and three levels of adaptivity. There were 16 fine grid cells spanning the nozzle radius. In Figure 8.11 we illustrate our computational results. The bubble diameters for the 2nd and 3rd bubbles were $4.86E-3m$ and $4.91E-3m$ respectively which is

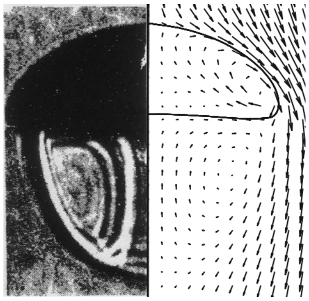

Fig. 8.9. *Terminal bubble shape for a gas bubble rising in a viscous liquid. The density ratio and viscosity ratio are 694:1 and 6557:1 respectively. The left result corresponds to experimental results from [5]. The right result corresponds to a bubble computed in a 3d axisymmetric coordinate system with* $\Delta t = 4.8E-4$ *(stable surface tension approach enabled).*

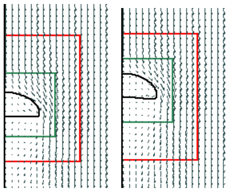

Fig. 8.10. *Terminal bubble shape for a gas bubble rising in a viscous liquid. The density ratio and viscosity ratio are 694:1 and 6557:1 respectively. Bubbles computed assuming a 3d coordinate system. The left result corresponds to* $\Delta t = 1.4E-3$, $t = 3.5$ *and the right result corresponds to* $\Delta t = 2.2E-4$, $t = 1.0$. *Plots represent the x-z slice of data at* $y = 0$.

in good agreement with the experimental result of $4.99E-3m$ and with our previous computational results of $4.85E-3m$ and $4.90E-3m$ respectively.

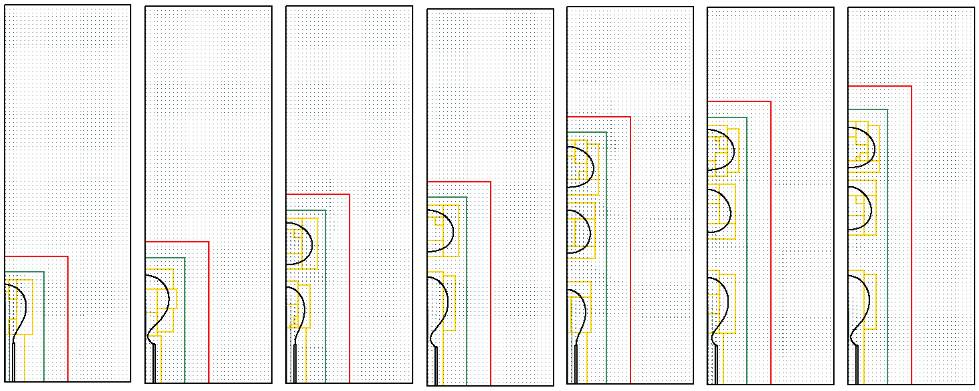

Fig. 8.11. *Bubble formation computed using our stable surface tension algorithm. Results computed using a 3d r-z coordinate system. The nozzle radius is* $8.5E-4m$, *the inflow velocity is* $0.44m/s$, *the density ratio is* $1015:1$ *and the viscosity ratio is* $6923:1$. *From left to right, top to bottom, results are plotted at dimensionless times of* $t = 32.7$, $t = 39.1$, $t = 63.6$, $t = 70.0$, $t = 94.5$, $t = 101.4$, *and* $t = 102.3$.

9. Conclusions. We have developed a method based on volume preserving motion by mean curvature which stabilizes two-phase computations in which there is surface tension related stiffness. With proper scaling of the Navier-Stokes equations, our method is unconditionally stable with respect to the surface tension force. Also, we have introduced a height function level set approach to discretizing curvature, which performs just as well as the height function volume-of-fluid approach to discretizing curvature, but avoids difficulties when two interfaces are in close proximity of each other. Our computational tests demonstrate good agreement between our results and previous computations and experiments. In particular, we have shown that for some problems, we can get a factor of 5 or more speed-up for the same accuracy. Lastly, our method was shown to be robust to complex interfaces which might merge or pinch-off.

REFERENCES

[1] J. Cohen and M. Jeroen Molemaker, *Practical simulation of surface tension flows*, in SIGGRAPH 2004 Conference Proceedings, August 2004, pp. SIGGRAPH Sketch, Feasible fluid, foliage, fog. http://www.siggraph.org/s2004/conference/sketches.

[2] C. Galusinski and P. Vigneaux, *On stability condition for bifluid flows with surface tension: Application to microfluidics*, J. Comput. Phys., 227(12) (2008), pp. 6140–6164.

[3] J. Helmsen, P. Colella, and E.G. Puckett, *Non-convex profile evolution in two dimensions using volume of fluids*, LBNL technical report LBNL-40693, Lawrence Berkeley National Laboratory, 1997.

[4] F. W. Helsby and K. R.Tuson, *Behaviour of air bubbles in aqueous solutions*, Research, 8 (1955), p. 270.

[5] J.G. Hnat and J.D. Buckmaster, *Spherical cap bubbles and skirt formation*, Physics of Fluids, 19 (2) (1976), pp. 182–194.

[6] J.I. Hochstein and T.L. Williams, *An implicit surface tension model, aiaa paper 96-0599*, in Proceedings of the 34th AIAA Aerospace sciences meeting and exhibit, Jan 15-18 1996. Reno, NV.

[7] T.Y. Hou, J.S. Lowengrub, and M.J. Shelley, *Removing the stiffness from interfacial flows with surface tension*, J. Comput. Phys., 114 (1994), p. 312.

[8] S. Hysing, *A new implicit surface tension implementation for interfacial flows*, Int. J. Numer. Meth. Fluids, 51(6) (2006), pp. 659–672.

[9] M. Kang, R. Fedkiw, and X.-D. Liu, *A boundary condition capturing method for multiphase incompressible flow*, J. Sci. Comput., 15 (2000), pp. 323–360.

[10] H. Lamb, *Hydrodynamics*, Dover Publications, New York, 1932.

[11] M. Raessi, M. Bussmann, and J. Mostaghimi, *A semi-implicit finite volume implementation of the csf method for treating surface tension in interfacial flows*, Int. J. Numer. Meth. Fluids, to appear, (2008).

[12] J.A. Sethian, *Level Set Methods and Fast Marching Methods: Evolving interfaces in Computational Geometry, Fluid Mechanics, Computer Vision and Science*, Cambridge, 1999.

[13] P.J. Slikkerveer, E.P. van Lohuizen, and S.B.G. O'Brien, *An implicit surface tension algorithm for Picard solvers of surface-tension-dominated free and moving boundary problems*, International Journal Numerical Methods in Fluids, 22 (1996), pp. 851–865.

[14] P.A. Stewart, N. Lay, M. Sussman, and M. Ohta, *An improved sharp interface method for viscoelastic and viscous two-phase flows*, Journal of Scientific Computing, 35(1) (2008), pp. 43–61.

[15] M. Sussman, *A second order coupled levelset and volume of fluid method for computing growth and collapse of vapor bubbles*, Journal of Computational Physics, 187 (2003), pp. 110–136.

[16] ———, *A parallelized, adaptive algorithm for multiphase flows in general geometries*, Computers and Structures, 83 (2005), pp. 435–444.

[17] M. Sussman and M. Ohta, *Improvements for calculating two-phase bubble and drop motion using an adaptive sharp interface method*, Fluid Dynamics and Materials Processing, 3(1) (2007), pp. 21–36.

[18] M. Sussman and E.G. Puckett, *A coupled level set and volume of fluid method for computing 3D and axisymmetric incompressible two-phase flows*, J. Comp. Phys., 162 (2000), pp. 301–337.

[19] M. Sussman, K.M. Smith, M.Y. Hussaini, M. Ohta, and R. Zhi-Wei, *A sharp interface method for incompressible two-phase flows*, J. Comp. Phys., 221(2) (2007), pp. 469–505.