# Direct Interface Tracking of Droplet Deformation 

Meizhong Dai, Haoshu Wang, J. Blair Perot, and David P. Schmidt

University of Massachusetts, Amherst

## Department of Mechanical and Industrial Engineering


#### Abstract

Direct interface tracking computes spray behavior based only on first principles. It is an advanced form of direct numerical simulation, but with the emphasis shifted from resolving details of turbulence to details of multiphase flow. The moving interface requires special treatment and advanced numerical methods. A code that is capable of accurate resolution of three-dimensional free-surface deformation has been constructed. The Navier-Stokes equations for the liquid phase are solved on a deforming unstructured mesh. This technique tracks the boundary precisely, similar to marker-and-cell methods. However the adaptive mesh deforms with the interface. Furthermore, this new method avoids the surface reconstruction required in volume of fluid methods. The numerical scheme produces a positive definite matrix that is solved using a conjugate gradient method. In order to maintain mesh quality, the mesh point connectivity and node locations are updated each time step. The results demonstrate the performance and accuracy of this technique. The method used for the surface tension force is


shown to be second-order accurate in space. By locally fitting the free surface to a parabola when evaluating curvature, problems with numerical noise in the solution are avoided. A time step criterion based on free surface numerical stability is discussed. The results for a deforming drop and collapsing ligament are presented. The code is validated by comparing to the theoretical period for drop deformation.

Keywords: free-surface, droplet, unstructured mesh, moving-mesh, interface tracking

## Nomenclature:

| $A_{\mathrm{f}}$ | Cell face area |
| :--- | :--- |
| $\mathrm{CV}, V, V_{\text {cell }}$ | Control volume |
| $d$ | Diameter |
| $\mathbf{f}$ | Body force |
| $\mathbf{h}$ | Free surface profile vector |
| $\mathbf{I}$ | The identity matrix |
| $k$ | Spring stiffness coefficient |
| $L e$ | Length of edges |
| $\hat{\mathbf{n}}$ | Unit normal vector |
| $n$ | Time step index, oscillation mode |
| $p$ | Pressure |
| $\mathbf{r}$ | Position vector |
| $\mathbf{r}_{\mathrm{f}}^{\mathrm{CG}}$ | Vector pointing from face center to cell center |


| $R$ | Radius of curvature |
| :--- | :--- |
| $\mathbf{s}$ | Stream function |
| $t$ | Time |
| $\hat{\mathbf{t}}$ | Unit tangent vector |
| $T$ | Period |
| $U$ | Face normal velocity |
| $\mathbf{u}$ | Cell velocity |
| $\mathbf{v}$ | Mesh velocity |
| $\varepsilon$ | Small ratio between undisturbed jet radius and characteristic axial length |
| $\sigma$ | Surface tension coefficient |
| $\phi$ | Velocity potential |
| $\psi, \psi$ | Stream function |
| $\rho$ | Density |
| $\mu$ | Viscosity |
| $\lambda$ | Wave length of a surface disturbance |
| $\nu$ | Dynamic viscosity |

## Introduction

The complexities of spray behavior are often very difficult to observe directly. Sprays usually evolve over small time and space scales. Furthermore, high number densities of droplets can impede optical access. For understanding basic spray physics, simulation based on first principles may be helpful. A model that relies only on the Navier-Stokes equations could generate trustworthy results that would provide complete detail about droplet behavior.

The field of turbulence has benefited from the analogue for single-phase flow: Direct Numerical Simulation (DNS). DNS has provided information for developing advanced turbulence models as well as for model validation. For two-phase flow, the calculation must include the ability to track the interface. The interface represents a discontinuity in fluid properties and a discontinuous stress due to surface tension. Thus special numerical techniques are required. One of the earlier attempts to simulate free-surface flows was Kothe et al. [1]. They created the RIPPLE code that used a linear stepwise reconstruction of the interface. Coverage of modern developments in interface tracking methods can be found in the reviews by Hyman [2] and Kothe [3]. As defined by Kothe, these methods fall into two categories: tracking methods, such as moving-mesh, front tracking, boundary integral and particle schemes, and capturing methods, such as continuum advection, volume tracking, level set, and phase field schemes.

The current work uses a moving-mesh method. While the moving-mesh method normally cannot equal Eulerian interface capturing methods in topological robustness, it does have the ability to solve the interface evolution directly, and does not have the smearing errors in the re-
construction of the interface shape, which are very common with some of the Eulerian methods as discussed in [3] and [4].

There have been only a few major attempts to apply interface tracking to primary atomization [5][6]. Recent schemes using moving-mesh methods include Welch [7] and Cristini et al. [8]. Welch simulated moderate deformation of a two-phase two-dimensional drop with a triangular mesh. Cristini et al. used an adaptive mesh-restructuring algorithm and a boundary element method (BEM) to simulate three-dimensional drop breakup and coalescence in creeping flow. Their adaptive algorithm was only applicable to the surface mesh. The current work solves the complete Navier-Stokes equations on a three dimensional unstructured mesh. The method, as described below, should be able to reveal physical details of the spray, though the present results are limited to solving one-phase flow with a free surface boundary condition.

## Governing Equations and Numerical Scheme

This work is an extension from Xing et al [9] and Nallapati et al [10]'s schemes in order to simulate large deformation in three-dimensions. Its numerical method is based on a streamfunction formulation of the Navier-Stokes equations. The complete Navier-Stokes equations are solved in two and three dimensions on a deforming unstructured mesh. The basic equations were solved for a deforming, moving, control volume, avoiding the interpolation errors inherent in global remeshing. The method has many common features with Arbitrary Lagrangian-Eulerian methods. However the current approach is not a fractional step method, in contrast to most ALE codes.

Xing et al [9] and Nallapati et al [10] have provided some information about the numerical scheme that will be used. An additional description is given here. Fig. 1 shows a twodimensional example of an unstructured mesh. This kind of mesh has no regularity and allows maximum flexibility in matching mesh cells with the boundary surfaces. For consistency with three-dimensions, each triangle will be the control volume and considered an infinitely long prism, so that the two-dimensional definitions of cells, faces, and edges will be in accordance with their three-dimensional counterparts. Here the basic parameter and real unknown is the stream function, which is a vector in both two- and three-dimensions (in two-dimensions the vector simply points along z-axis). The stream function is located on edge centers. By using a stream function one avoids solving the continuity equation. To construct the velocity field (which is defined at each cell center) from the stream function, there are two steps: first construct face normal velocities, $U$, from the stream function using Stokes' theorem. The face normal velocities can be calculated from stream function, $\mathbf{s}$, as:

$$
\begin{equation*}
\int_{\text {face }} U d A=\int_{\text {face }} \mathbf{u} \cdot \hat{\mathbf{n}} d A=\int_{\text {face }}(\nabla \times \mathbf{s}) \cdot \hat{\mathbf{n}} d A \tag{1}
\end{equation*}
$$

Then using Stokes' theorem:

$$
\begin{equation*}
\int_{\text {face }}(\nabla \times \mathbf{s}) \cdot \hat{\mathbf{n}} d A=\oint_{\substack{\text { all edges } \\ \text { of the face }}} \mathbf{s} \cdot \hat{\mathbf{t}} d l \tag{2}
\end{equation*}
$$

Therefore the face fluxes can be obtained in the form of :

$$
\begin{equation*}
U A_{\mathrm{f}}=\sum_{\substack{\text { all edges } \\ \text { of the face }}} \quad \mathbf{s} \cdot \hat{\mathbf{t}} \Delta l \tag{3}
\end{equation*}
$$

This equation shows that only one component of the stream function, $\mathbf{s} \cdot \mathbf{t}$, is really needed. Therefore only the stream function's projection along the edges need to be calculated and updated, so that it only needs to be stored as a scalar in code implementation. Also solving only
one component of the unknown vector reduces the size of the final matrix. The second step is to construct the velocities at the cell centers from the face normal velocities $U$. Because $\nabla \mathbf{r}=\mathbf{I}$ and $\nabla \cdot \mathbf{u}=0$, one has:

$$
\begin{equation*}
\overline{\mathbf{u}} V_{\text {cell }}=\int_{\text {cell }} \mathbf{u} d v=\int_{\text {cell }}[(\nabla \mathbf{r}) \cdot \mathbf{u}+(\nabla \cdot \mathbf{u}) \mathbf{r}] d v \tag{4}
\end{equation*}
$$

Next from the product rule, the above form can be rewritten as:

$$
\begin{equation*}
\int_{\text {cell }}[(\nabla \mathbf{r}) \cdot \mathbf{u}+(\nabla \cdot \mathbf{u}) \mathbf{r}] d v=\int_{\text {cell }} \nabla \cdot\left(\mathbf{r u}^{\mathrm{T}}\right) d v \tag{5}
\end{equation*}
$$

Then Gauss' Divergence Theorem is applied, and the integral is approximated with face-averaged values. The variable $\overline{\mathbf{r}}$ denotes the location of the face center.

$$
\begin{align*}
& \int_{\text {cell }} \nabla \cdot\left(\mathbf{r} \mathbf{u}^{\mathrm{T}}\right) d v=\oint_{\text {face }} \mathbf{r} \mathbf{u}^{\mathrm{T}} \cdot \hat{\mathbf{n}} d s \approx \sum_{\text {cell faces }} \oint_{\text {each face }} \overline{\mathbf{r} u} \cdot \hat{\mathbf{n}} d s=\sum_{\text {cell faces }} \overline{\mathbf{r}} \oint_{\text {each face }} \mathbf{u} \cdot \hat{\mathbf{n}} d s=\sum_{\text {cell faces }} \overline{\mathbf{r}} U A_{\mathrm{f}}(6) \\
& \mathbf{u}=\frac{1}{V_{\text {cell }}} \sum_{\text {cell faces }} U \mathbf{r}_{\mathrm{f}}^{\mathrm{CG}} A_{\mathrm{f}} \tag{7}
\end{align*}
$$

Here $V_{\text {cell }}$ is the cell volume and $\mathbf{r}_{\mathrm{f}}{ }^{\mathrm{CG}}$ is the vector pointing from a face circumcenter to the cell circumcenter. Actually $\mathbf{r}_{f}^{\mathrm{CG}}$ could be a vector pointing from any common point to the face center. The conservation properties of this technique for solving the Navier-Stokes equations are proven by Perot [11].

Momentum equations can be developed after the velocities are obtained. The procedure is complex since the mesh has to be moved in order to track the free surface. Beginning with the Reynolds transport theorem for incompressible flow, one has:

$$
\begin{equation*}
\frac{d}{d t} \int_{\mathrm{CV}} \rho \mathbf{u} d v+\int_{\mathrm{CV}} \nabla \cdot \rho \mathbf{u}(\mathbf{u}-\mathbf{v}) d v=\int_{\mathrm{CV}} \rho \mathbf{f} d v-\int_{\mathrm{CV}} \nabla p d v+\int_{\mathrm{CV}} \mu \nabla^{2} \mathbf{u} d v \tag{8}
\end{equation*}
$$

where $\mathbf{f}$ is the body force, $\mathbf{v}$ is the mesh velocity, and CV is the control volume. Assuming the control volume is small enough and the velocity $\mathbf{u}$ is linear within each of them, the integral form can be changed to:

$$
\begin{align*}
& \frac{1}{\Delta t}\left[(\rho \mathbf{u} V)^{n+1}-(\rho \mathbf{u} V)^{n}\right]+(\nabla \cdot \rho \mathbf{u}(\mathbf{u}-\mathbf{v}) V)^{n}=(\rho \mathbf{f} V)^{n}-(\nabla p V)^{n+1}+\left(\mu \nabla^{2} \mathbf{u} V\right)^{n}  \tag{9}\\
& \frac{1}{\Delta t}\left[(\rho \mathbf{u})^{n+1}-(\rho \mathbf{u})^{n} \frac{V^{n}}{V^{n+1}}\right]+(\nabla \cdot \rho \mathbf{u}(\mathbf{u}-\mathbf{v}))^{n} \frac{V^{n}}{V^{n+1}}=(\rho \mathbf{f})^{n} \frac{V^{n}}{V^{n+1}}-\nabla p^{n+1}+\left(\mu \nabla^{2} \mathbf{u}\right)^{n} \frac{V^{n}}{V^{n+1}} \tag{10}
\end{align*}
$$

where the parameters $\mathbf{u}, p$, and $\rho$ are defined at the center of each control volume, and $n$ and $\mathrm{n}+1$ represent different time steps. The value of the cell volume $V^{n+1}$ at the $n+1$ time step is calculated with a predictor-corrector scheme. Crank-Nicolson differencing is applied to the diffusion terms. The convection terms are solved explicitly. The divergence operators in the convection and diffusion terms can be evaluated using Gauss' Divergence Theorem:

$$
\begin{align*}
& {[\nabla \cdot \rho \mathbf{u}(\mathbf{u}-\mathbf{v})]=\frac{1}{V} \sum_{\text {cell faces }} \rho \mathbf{u}(\mathbf{u}-\mathbf{v}) \cdot \hat{\mathbf{n}} A_{\mathrm{f}}}  \tag{11}\\
& {[\mu \nabla \cdot(\nabla \mathbf{u})]=\frac{1}{V} \sum_{\text {cell faces }} \mu \nabla \mathbf{u} \cdot \hat{\mathbf{n}} A_{\mathrm{f}}} \tag{12}
\end{align*}
$$

As a final step, the velocity $\mathbf{u}$ is expressed in terms of the stream function component $\mathbf{s} \cdot \hat{\mathbf{t}}$ defined on each edge using Eqns. (3) and (7). Then Eqn. (10) is integrated around the edges, i.e. along the route marked by arrows in Fig. 1. This procedure is like using a discrete curl operator on the systems of Eqn. (10), and this curl operation will eliminate the pressure gradients throughout the domain except on the boundaries. Now one has a linear system of $n$ equations and n unknowns ( n is the number of edges in the whole domain). New time step values are obtained by integrating this system with a third order Runge-Kutta method. The resulting equations form a positive semi-definite matrix. With these conditions met, a conjugate gradient method can be used to solve the equations efficiently.

## Surface Tension Evaluation

A constant pressure boundary condition is used for space outside the free surface, and surface tension is treated as an additional term in the boundary cells' pressure:

$$
\begin{equation*}
p^{\prime}=p+\sigma\left(\frac{1}{R_{1}}+\frac{1}{R_{2}}\right) \tag{13}
\end{equation*}
$$

In this equation the normal viscous stresses are neglected because the Capillary Number is much less than unity, as discussed in [12]. The values of $\mathrm{R}_{1}$ and $\mathrm{R}_{2}$ are the radii of curvature in two orthogonal directions along the interface. The sum of the reciprocals of $R_{1}$ and $R_{2}$ will be denoted as $1 / R_{\text {effective }}$, so that the equations may be generally represented for two or three dimensions. Nallapati et al [10] treated surface tension as the sum of the surface forces on each edge of a cell face using following equation:

$$
\begin{equation*}
\frac{1}{R_{\text {effective }}}=-\hat{\mathbf{n}} \cdot \nabla^{2} \mathbf{h}(x, y) \tag{14}
\end{equation*}
$$

The vector $\mathbf{h}$ represents the surface as a function of $x$ and $y$. First the surface tension forces on each edge are obtained from the gradients of neighboring face center coordinates, and then they are sum to each boundary face. This method has the attractive property that the total surface tension forces, when summed over a closed two-dimensional curve or a three-dimensional surface, vanish identically. So when surface tension is integrated over a droplet, there is no residual surface tension imbalance that would accelerate the drop center of mass. Though this method is accurate in two dimensions, it suffers from numerical noise in three dimensions. Fig. 2 shows that although the average value from Eqn. (14) is around the analytical value, its distribution is so widely scattered that it inundates all the solution caused by physical factors.

When a curl operation is applied on the free-surface boundary, one gets large amount of numerical noise. The Laplacian operator in Eqn. (14) can also be calculated as the sum of average unit tangential vectors on each edge over each boundary face, which yields similar results. This three-dimensional shortcoming is a consequence of taking higher-order derivatives of unequally spaced points. In two dimensions, the problem is not particularly severe, because mesh generation tends to precisely space points at equal intervals along the interface.

In this work curvature in three dimensions is calculated on each boundary face by a new surface fitting method. Zinchenko et al. [13] used a parabolic fit for surface tension, calculating curvature at nodes rather than faces. Their algorithm was necessarily very complicated. The current implementation is much simpler, because the force is defined at the face center. First, since the curvature is independent of the coordinate system, one can set up a local coordinate system whose $z$-axis coincides with the face normal direction. This step is necessary to avoid an ill-conditioned matrix. In three dimensions, the local surface of a cell face is fitted by:

$$
\begin{equation*}
z(x, y)=f(x, y)=a_{6} x^{2}+a_{5} y^{2}+a_{4} x y+a_{3} x+a_{2} y+a_{1} \tag{15}
\end{equation*}
$$

and coefficients $a_{1}$ through $a_{6}$ are obtained from the coordinates of six nodes. Three nodes are from the triangular face itself and three nodes are from the three neighboring faces, producing a six by six linear system. This dependence is ideal, because it produces a template that is centered around the face of interest. Then the local curvature is:

$$
\begin{equation*}
\frac{1}{R_{\text {effective }}}=-\frac{1}{2}\left[\left(1+\mathrm{f}_{\mathrm{y}}^{2}\right) \mathrm{f}_{\mathrm{xx}}-2 \mathrm{f}_{\mathrm{x}} \mathrm{f}_{\mathrm{y}} \mathrm{f}_{\mathrm{xy}}+\left(1+\mathrm{f}_{\mathrm{x}}^{2}\right) \mathrm{f}_{\mathrm{yy}}\right] /\left[1+\mathrm{f}_{\mathrm{x}}^{2}+\mathrm{f}_{\mathrm{y}}^{2}\right]^{3 / 2} \tag{16}
\end{equation*}
$$

Fig. 2 shows the comparison between Eqn. (14) and the curve fitting method. It is clear that the latter method has much less noise, which is essential for stability. Curve fitting slightly overpredicts the surface curvature, since a parabola is used to find the radius of a supposed sphere.

Fig. 3 shows the numerical results of surface tension calculation for a perfect sphere of diameter 2 mm . It shows that the curve fitting method has second order accuracy, which is expected from the fact that a quadratic function is used to fit the surface.

## Mesh-Moving Scheme

This work uses the adaptive mesh approach to track the free surface. The mesh nodes on the free surface are moved in a Lagrangian manner:

$$
\begin{equation*}
\dot{\mathbf{x}}=\mathbf{u}_{\text {node }} \tag{17}
\end{equation*}
$$

while the interior nodes are moved using a spring analogy with zero equilibrium length, which simply treats each cell edge (or cell face in two dimensions under the definitions in Fig. 1) as if it were under tension proportional to its length. The resultant forces at each node are computed from the tensions to adjust its position:

$$
\begin{equation*}
\mathbf{x}_{i}{ }^{n+1}-\mathbf{x}_{i}{ }^{n}=\sum_{j: \text { neighbor nodes }} k\left(\mathbf{x}_{i j}{ }^{n}-\mathbf{x}_{i}{ }^{n}\right), i=1, \ldots, \text { number of total nodes. } \tag{18}
\end{equation*}
$$

Here $k$ is a numerical stiffness coefficient that controls how fast the mesh relaxes. The spring analogy has been used by others [14][15][16], with several choices for $k$. In this work, a constant value of $k$ is sufficient to produce a good quality mesh. Also the advantage of this form of $k$ is that one can use an implicit method for the right-hand side in Eqn. (18), resulting in a system of linear equations that are positive definite and hence can also be solved by conjugate gradient. A value of $k$ on the order of unity is found to be satisfactory.

A familiar difficulty with the Lagrangian scheme is that the distribution of the surface nodes quickly becomes irregular and leads to an unusable mesh [13]. To circumvent this difficulty a
similar scheme is also used on free surface nodes, in which these nodes are smoothed while constrained to remain on the surface:

$$
\begin{equation*}
\mathbf{x}_{i}{ }^{n+1}-\mathbf{x}_{i}{ }^{n}=(\mathbf{I}-\hat{\mathbf{n}} \hat{\mathbf{n}}) \sum_{j: \text { neighbor nodes }} k\left(\mathbf{x}_{i j}{ }^{n}-\mathbf{x}_{i}{ }^{n}\right), i=1, \ldots, \text { number of free surface nodes } \tag{19}
\end{equation*}
$$

In this equation, all the terms related to $\hat{\mathbf{n}} \hat{\mathbf{n}}$ are non-linear and therefore are solved explicitly. A value of $k$ on the order of 0.01 is found enough to spread the surface nodes evenly.

When the mesh evolution produces some poorly shaped cells, edge-swapping methods are used to change the local connectivity. In two dimensions the preferred choice would be Lawson's Delaunay-based algorithm [17]. It successively examines each pair of neighboring triangles. If the two triangles lose their Delaunay property, their common face can just be flipped to form two good ones as shown in Fig. 4. In two dimensions, the Delaunay criteria can be shown to be equivalent to maximizing the minimum angle [18], therefore this algorithm can produce an almost equilateral mesh.

Similar transformations can be achieved in three dimensions through face- and edge- swapping. Face-swapping is shown in Fig. 5. A pair of neighboring tetrahedra can be split to form three new tetrahedra (2-3 swapping), or vice versa: three tetrahedra sharing one common edge can be combined to form a new pair of neighboring tetrahedra (3-2 swapping). Edge-swapping is shown in Fig. 6. The left side shows the original four tetrahedra (AB14, AB12, AB34, and AB23) sharing edge AB , which is perpendicular to the paper. If polyhedron AB 124 is concave on edge AB , it can be re-configured to form four new tetrahedra 42A3, 42B3, 421A, and 421B. For the case when more than four tetrahedra share one common edge, multiple possible configurations exist and a test must be performed to decide which configuration to use. Note that this process
can be implemented by a series of 2-3 and one final 3-2 face-swappings, by examining each pair of neighboring cells around AB .

In three dimensions, the Delaunay criterion is not always sufficient to guarantee good cells. Swapping occasionally produces cells that are very flat, with their four vertices nearly coplanar. These cells, called "sliver cells" are a known defect of using the Delaunay criteria [19]. Sliver cells can lead to large discretization errors and instability. They also impose a new difficulty for the smoothing methods discussed before: some sliver cells are so flat that they could easily be inverted by smoothing methods and result in negative volumes. Consequently, a minimummaximum dihedral angle criterion is used instead of the Delaunay criterion in three dimensions. Although the dihedral angle criterion doesn't guarantee a global optimum, it works much better than the Delaunay criterion.

## Stability

In addition to the CFL limit, a criterion of free surface numerical stability is required. The mixed character of the flow at the surface makes this difficult. To derive a stability relation, the momentum normal to the interface was considered. For simplicity, the viscous and convective acceleration terms were dropped. Viscosity is treated implicitly in this work, and the convective terms would give rise to the CFL limit. The simplified situation is potential flow driven by surface tension. This is the well-known case of the propagation of small waves in deep water with negligible gravity, also known as capillary waves. Sinusoidal capillary waves travel at a speed, $c$, that depends on the wavelength of the disturbance $\lambda$ [20].

$$
\begin{equation*}
c=\sqrt{\frac{2 \pi \sigma}{\rho \lambda}} \tag{20}
\end{equation*}
$$

The value of $\lambda$ would be approximately the same as the mesh resolution, assuming that the instability manifests itself at the highest resolvable frequency. This assumption results in propagation with a wave speed of $\sqrt{\frac{2 \pi \sigma}{\rho \Delta x}}$. Thus by the CFL criterion, the surface will be stable for an appropriate explicit scheme when:

$$
\begin{equation*}
\Delta t<C \sqrt{\frac{\rho \Delta x^{3}}{2 \pi \sigma}} \tag{21}
\end{equation*}
$$

This result is the same as Eqn. (61) in Brackbill et al [21], which is used for a continuum surface force scheme. The constant C in Eqn. (21) is of order unity and depends on the numerical method. Eqn. (21) provides a way of predicting a constraint on the time step in free surface calculations. The validity of this deduction is shown for the moving mesh method in the following section.

## Results

A test calculation was used to check the prediction of Eqn. (21). A two-dimensional circle, representing the cross section of an infinite cylinder, was simulated for various values of surface tension and a range of mesh resolutions. This test case was identical to the two-dimensional validation discussed later. The flowfield was subjected to a small initial perturbation and checked for stability. The time stepping method was a three step second-order Runge-Kutta method. A von Neumann analysis of this Runge-Kutta method indicates that the value of $C$ should be exactly two. The results plotted in Fig. 7 show that Eqn. (21) correctly predicts the
stability limits and the dependency on fluid properties. This is confirmed in Fig. 8, where the mesh resolution is varied and the properties are held constant.

For a simple two-dimensional test case, the oscillation of a circular cylinder was calculated using 2,600 cells. The initial velocity potential was $\phi=0.025 r^{2} \cos (2 \theta)=0.05\left(x^{2}-y^{2}\right)$, the stream function was $\psi=0.1 x y$, and the initial velocity field was $v_{x}=0.1 x, v_{y}=-0.1 y$. According to Lamb [22] the oscillation period for this mode $n=2$ perturbation is:

$$
\begin{equation*}
T=2 \pi\left[n\left(n^{2}-1\right) \frac{\sigma}{\rho r_{0}^{3}}\right]^{-0.5}=2 \pi\left[6 \frac{\sigma}{\rho r_{0}^{3}}\right]^{-0.5} \tag{22}
\end{equation*}
$$

Table 1 shows the numerical result of a liquid cylinder oscillation given a small perturbation. It is clear that the two-dimensional numerical results are very close to the theoretical values. For a larger perturbation, which demonstrates the function of mesh smoothing and flipping, images are shown in Fig. 9. A coarse mesh of 430 cells was used for the figure so that the cells could be clearly seen.

| $\sigma / \rho\left(\mathrm{m}^{3} / \mathrm{s}^{2}\right)$ | $10^{-8}$ | $10^{-7}$ | $10^{-6}$ | $10^{-5}$ | $7.56 \times 10^{-5}$ |
| :---: | :---: | :---: | :---: | :---: | :---: |
| $T(\mathrm{~s})$ : Theoretical value | 0.81115 | 0.25651 | 0.08112 | 0.02565 | $9.3291 \times 10^{-3}$ |
| $T(\mathrm{~s})$ : Test value | 0.81111 | 0.25650 | 0.08111 | 0.02560 | $9.3290 \times 10^{-3}$ |
| Percentage Error | $-0.0049 \%$ | $-0.0039 \%$ | $-0.0123 \%$ | $-0.1949 \%$ | $-0.0011 \%$ |

Table 1: Period of liquid cylinder oscillation. Initial parameters are: $r_{0}=\mathbf{0 . 0 0 1 m}$, $v=1.781 \times 10^{-6} \mathrm{~m}^{2} / \mathrm{s}$. A fine mesh of 2,600 cells was used.

As a three-dimensional test case, droplet oscillation was calculated using 4,500 cells. The initial velocity potential was $\phi=0.25 r^{2} \cos (2 \theta)=0.5\left(x^{2}+y^{2}-2 z^{2}\right)$, the stream function $\boldsymbol{\psi}=1.5 y z \boldsymbol{i}-$ $0.5 x z j+0.5 y x \boldsymbol{k}$, and the initial velocity field was $v_{x}=x, v_{x}=y, v_{z}=-2 z$. According to Lamb [22] the drop oscillation period for this mode $n=2$ perturbation is:

$$
\begin{equation*}
T=2 \pi\left[n(n-1)(n+2) \frac{\sigma}{\rho r_{0}^{3}}\right]^{-0.5}=2 \pi\left[8 \frac{\sigma}{\rho r_{0}^{3}}\right]^{-0.5} \tag{23}
\end{equation*}
$$

Table 2 shows the numerical results of the liquid drop oscillation calculation. Because the magnitude of the perturbation used in three dimensions is larger than two dimensions and the theoretical value is only a linearized approximation, the percentage errors become larger. For large distortion, the results are shown in Fig. 10, obtained with a coarse mesh of 1,300 cells.

| $\sigma / \rho\left(\mathrm{m}^{3} / \mathrm{s}^{2}\right)$ | $10^{-8}$ | $10^{-7}$ | $10^{-6}$ | $10^{-5}$ | $7.56 \times 10^{-5}$ |
| :--- | ---: | ---: | ---: | ---: | ---: |
| $T(s):$ Theoretical value | 0.702 | 0.222 | 0.0702 | 0.0222 | 0.00808 |
| $T(s):$ Test value | 0.715 | 0.228 | 0.0705 | 0.0223 | 0.00800 |
| Percentage Error | $1.8519 \%$ | $2.7027 \%$ | $0.4274 \%$ | $0.4505 \%$ | $-0.9901 \%$ |

Table 2: Liquid drop oscillation period value. Initial parameters are: $r_{0}=\mathbf{0 . 0 0 1 m}$, $v=1.781 \times 10^{-6} \mathrm{~m}^{2} / \mathrm{s}$. A fine mesh of 4,500 cells was used.

This numerical technique is also very useful for the case of a collapsing ligament, a flow which is almost entirely surface tension driven. The initial condition for this case was a ligament with a length of six diameters, plus two hemispherical ends. The collapse of the ligament is shown below in Fig. 11.

## Conclusions

A numerical method for calculating free surface distortion has been described, examined, and demonstrated. The method solves the Navier-Stokes equations in deforming, moving, volumes. Cell faces on the interface move with the free-surface. The method is designed to work in both two and three dimensions using triangular and tetrahedral control volumes, respectively. The resulting matrices are positive semidefinite and can be solved efficiently using a conjugate gradient method.

This approach has the advantage of exact surface tracking, allowing direct calculation of surface tension forces. Different methods of calculating the surface curvature were investigated. A method of fitting a parabola to the surface allows second-order accuracy without excessive amounts of numerical noise. A stability analysis correctly predicted that the largest stable time step corresponds to $\Delta \mathrm{x}^{-3 / 2}$.

The results show the tremendous potential of this method. The numerical method described here could be used as easily for simulating liquid films, sloshing, and other free surface flows. The primary impediment is the degeneration of mesh quality as the liquid deforms. In twodimensions, mesh smoothing and flipping can be used to properly update the mesh point locations and connectivity. In three dimensions, the maximum-minimum dihedral angle criterion is more efficient than Delaunay property. With the ability to handle unlimited distortion, this approach would provide an excellent technique for calculating free-surface behavior. Future work will address multi-phase flow and changing topology.

## References

[1]D. B. Kothe, R. C. Mjolsness, and M. D. Torrey, Ripple: A Computer Program for Incompressible Flows with Free Surfaces, Technical Report LA-12007-MS, Los Alamos National Laboratory, 1991.
[2]J. M. Hyman, Numerical Methods for Tracking Interfaces, Physics D, 12, 396, 1984.
[3]D. B. Kothe, Perspectives on Eulerian Finite Volume Methods for Incompressible Interfacial Flows, Free Surface Flows, edited by H. C. Kuhlmann and H.-J. Rath, Springer-Verlag, New York.
[4]W. J. Rider, D. B. Kothe, S. J. Mosso and J. H. Cerutti, Accurate Solution Algorithms for Incompressible Multiphase Flows, AIAA-95-0699, 1995.
[5]B. Lafaurie, T. Mantel and S. Zaleski, Direct Numerical Simulation of Liquid Jet Atomization, Third International Conference on Multiphase Flow, ICMF'98, Lyon, France, June 8-12,1998.
[6]W. Tauber and G. Trygvasson, Primary Atomization of a Jet, Proc. of ASME FED Summer Meeting, Boston, 2000.
[7]S. W. J. Welch, Local Simulation of 2-Phase Flows Including Interface Tracking with MassTransfer, J Comp. Phys, 121 (1): 142-154 OCT 11995.
[8]V. Cristini, J. Blawzdziewicz and M. Loewenberg, An Adaptive Mesh Algorithm for Evolving Surfaces: Simulations of Drop Breakup and Coalescence, J Comp. Phys 168(2): 445463 APR 102001.
[9]X. Zhang, D. Schmidt and B. Perot, Accuracy and Conservation Properties of a ThreeDimensional Unstructured Staggered Mesh Scheme for Fluid Dynamics, J Comp. Phys, 175: 764-791, 2002.
[10]R. Nallapati and J. B. Perot, Numerical Simulation of Free-Surface Flows Using a Moving Unstructured Mesh, Proceedings of ASME FEDSM'00.
[11] J. B. Perot, Conservation Properties of Unstructured Staggered Mesh Schemes, J Comput. Phys. 159, 58-59(2000).
[12]A. Mazouchi, G. M. Homsy, Thermocapillary migration of long bubbles in cylindrical capillary tubes, Physics of Fluids, 12 (3): 542-549 Mar 2000.
[13] Alexander Z. Zinchenko, M. A. Rother, and R. H. Davis, A Novel Boundary-Integral Algorithm for Viscous Interaction of Deformable Drops, Phys. Fluids, 9 (6), 1997.
[14]J. T. Batina, Unsteady Euler Airfoil Solutions Using Unstructured Dynamic Meshes, AIAA Journal, 28 (8): 1381-1388 AUG 1990.
[15]B. Palmerio, An Attraction Repulsion Mesh Adaption Model For Flow Solution On Unstructured Grids, Computers \& Fluids, 23 (3): 487-506 MAR 1994.
[16] K. P. Singh, J. C. Newman, O. Baysal, Dynamic Unstructured Method For Flows Past Multiple Objects In Relative Motion, AIAA Journal, 33 (4): 641-649 APR 1995.
[17] C. L. Lawson, Transforming Triangulations, Discrete Math, 3, 365-372, 1972.
[18]C. L. Lawson, Software for C ${ }^{1}$ Surface Interpolation, Mathematical Software III, Page 161, edited by J. R. Rice (Academic Press, New York, 1977).
[19]M. Bern and D. Eppstein, Mesh Generation and Optimal Triangulation, Tech. Rep. CSL-921, Xerox PARC, 1992. Computing in Euclidean Geometry, D. Z. Du and F. K. Hwang, eds., World Scientific, 1992, pp. 23-90.
[20]I. G. Currie, Fundamental Mechanics of Fluids, 1974, McGraw-Hill, New York, pp. 186188.
[21]J. U. Brackbill, D. B. Kothe, C. Zemach, A Continuum Method For Modeling SurfaceTension, J Comp. Phys, 100 (2): 335-354 Jun 1992.
[22]H. Lamb, Hydrodynamics, 1945, 6th edition, Dover Publications, Inc.

## List of figures:

Figure 1: A two-dimensional unstructured mesh. Mesh elements are shown as labeled. The arrows show the route through which cell terms are integrated into vorticity terms

Figure 2: Curvature calculation of a sphere $(r=1 \mathrm{~mm})$, with curvature $1 / R_{\text {effective }}=2000$.

Figure 3. The convergence of the curve fitting scheme. Fitting the neighboring points to a parabolic surface shows second order convergence.

Figure 4. Edge-swapping in two dimensions.

Figure 5: Face-swapping in three dimensions: 2-3 or 3-2 swapping

Figure 6: Edge-swapping in three dimensions

Figure 7. Evaluation of the largest stable time step. The analytical result is Eqn. (21) with the constant set to 2.0. The liquid properties are varied and the mesh resolution is held constant.

Figure 8. Evaluation of the largest stable time step. The Analytical Value is from Eqn. (21) with the constant set to 2.0 . The properties are held constant and the mesh resolution is varied.

Figure 9. A two-dimensional planar calculation showing the function of mesh flipping and smoothing. A coarse mesh of 430 cells was used.

Figure 10: Liquid drop oscillation subject to a large disturbance. A coarse mesh of 1,300 cells was used.

Figure 11: Liquid drop oscillation subject to a large disturbance. The images proceed in time from left to right, and top to bottom.


Figure 1: A two-dimensional unstructured mesh. Mesh elements are shown as labeled. The arrows show the route through which cell terms are integrated into vorticity terms


Figure 2: Curvature calculation of a sphere ( $r=1 \mathrm{~mm}$ ), with curvature $\mathbf{1} / \boldsymbol{R}_{\text {effective }}=\mathbf{2 0 0 0}$.


Figure 3. The convergence of the curve fitting schemes. Fitting the neighboring points to a parabolic surface shows second order convergence to the analytical value).


Figure 4. Edge-swapping in two dimensions.


Figure 5: Face-swapping in three dimensions: 2-3 or 3-2 swapping


Figure 6: Edge -swapping in three dimensions


Figure 7. Evaluation of the largest stable time step. The analytical result is Eqn. (21) with the constant set to $\mathbf{2 . 0}$. The liquid properties are varied and the mesh resolution is held constant.


Figure 8. Evaluation of the largest stable time step. The Analytical Value is from Eqn. (21) with the constant set to 2.0 . The properties are held constant and the mesh resolution is varied.


Figure 9. A two-dimensional planar calculation showing the function of mesh flipping and smoothing. A coarse mesh of 430 cells was used.


Figure 10: Liquid drop oscillation subject to a large disturbance. A coarse mesh of $\mathbf{1 , 3 0 0}$ cells was used.


Figure 11: Liquid drop oscillation subject to a large disturbance. The images proceed in time from left to right, and top to bottom.

