JOURNAL OF COMPUTATIONAL PHYSICS **130,** 269–282 (1997)
ARTICLE NO. CP965590

# A High-Order Projection Method for Tracking Fluid Interfaces in Variable Density Incompressible Flows

Elbridge Gerry Puckett,* Ann S. Almgren,† John B. Bell,† Daniel L. Marcus† and William J. Rider‡

**Department of Mathematics and Institute of Theoretical Dynamics, University of California Davis, Davis, California 95616; †Center for Computational Sciences and Engineering, Lawrence Berkeley National Laboratory, Berkely, California 94720; ‡Los Alamos National Laboratory, Los Alamos, New Mexico 87545*

Received February 16, 1995; revised September 16, 1996

We present a numerical method for computing solutions of the incompressible Euler or Navier–Stokes equations when a principal feature of the flow is the presence of an interface between two fluids with different fluid properties. The method is based on a second-order projection method for variable density flows using an "approximate projection" formulation. The boundary between the fluids is tracked with a second-order, volume-of-fluid interface tracking algorithm. We present results for viscious Rayleigh–Taylor problems at early time with equal and unequal viscosities to demonstrate the convergence of the algorithm. We also present computational results for the Rayleigh–Taylor instability in air-helium and for bubbles and drops in an air–water system without surface tension to demonstrate the behavior of the algorithm on problems with large density and viscosity contrasts. © 1997 Academic Press

## 1. INTRODUCTION

Fluid flows with free surfaces or material interfaces occur in a large number of natural and technological processes. Casting processes, mold filling, thin film processes, extrusion, spray deposition, and fluid jetting devices are just a few of the areas in which material interfaces occur in industrial applications. Often properties that depend on the shape of the interface itself (e.g., surface tension) play an important role in the dynamics of the problem, and the physics (e.g., capillarity) can drive the flow making it essential to accurately determine the position, curvature, and topology of the interface. Numerical simulations are, in principle, ideally suited to study these flows; not only can the data be much more easily gathered but various physical processes can be turned on and off at will. In practice, however, the numerical computation of flows with material interfaces is difficult, especially if the interface separates fluids of dramatically different densities such as air and water.

There are two basic approaches that one can use to approximate flows with a material interface: "capturing" and "tracking." In interface capturing, the interface is treated as a region of steep gradient in some quantity (e.g., density $\rho$) that satisfies an advection equation of the form

$$\rho_t + \nabla \cdot (\rho \mathbf{u}) = 0, \tag{1}$$

where $\mathbf{u}$ denotes the fluid velocity. To obtain high-order accuracy one typically approximates solutions of (1) with an advection algorithm that is high-order in smooth regions and subject to some sort of monotonicity constraint (e.g., see [6, 14, 30]). This approach has been successfully used by Bell and Marcus to study a variety of problems [7, 32]. The primary advantages of this approach are that it is easy to implement and no additional algorithmic details are required to model topological changes of the interface. However, the front diffuses over several computational zones, resulting in a corresponding loss of accuracy (see, e.g., [19]).

In an interface tracking method the interface is treated explicitly as a sharp discontinuity moving through the grid. Tracking can offer better accuracy than capturing, but at the cost of greater algorithmic complexity. There are basically three possible representations of an interface in an interface tracking method:

- a piecewise polynomial function ("front tracking" and boundary integral methods)
- the level set of some function (level set methods)
- a collection of volume fractions (volume-of-fluid methods).

Each of these three approaches has its strengths and weaknesses. For example, the advantage of using a piecewise polynomial method is that one can design arbitrarily high-order accurate approximations to a smoothly varying front. This intrinsic accuracy can often allow one to resolve fine scale features of the front on relatively coarse underlying grids. Examples of piecewise polynomial methods include boundary integral methods (e.g., [3, 37, 50]), the front tracking methods of Glimm *et al.* [17, 18] and the algorithm developed by Tryggvason [61]. The piecewise polynomial approach has several weaknesses: it is relatively difficult to extend a piecewise polynomial method to three-dimensions (e.g., see [20]), one must design special procedures for handling the changes in the front topology, and it is not intrinsically conservative. (We note that Tryggvason and his coworkers have developed a three-dimensional piecewise polynomial method [61] and report that the mass loss is negligible for the flows they have studied [60].)

The level set approach was originally introduced by Osher and Sethian [38] and has since been applied to a wide variety of problems including bubbles and drops [54], Rayleigh–Taylor instability [34], flow by mean curvature [48], and dendritic growth and solidification [49]. In a level set method the interface is represented as the level set of some function $\phi$. This approach has two inherent strengths. One very useful feature of level set methods is that the representation of the interface as the level set of some function $\phi$ leads to convenient formulas for the interface normal and curvature. Another advantage of this approach is that no special procedures are required in order to model topological changes of the front. One simply updates the value of the level set function $\phi$ at each time step, and the location of the front at the new time is simply the zero contour of $\phi$.

In a volume-of-fluid method one tracks the volume of each material in cells that contain a portion of the interface, rather than the interface itself. At each time these volumes are used to reconstruct an approximation to the interface and this approximate interface is then used to update the values of the fluid volumes in each cell at the next time. A volume-of-fluid interface tracking method consists of two parts: an *interface reconstruction algorithm* for determining an approximation to the interface from a given collection of volumes and a volume-of-fluid *advection algorithm* or *transport algorithm* for determining the values of the fluid volumes at the new time from a given velocity field and the reconstructed interface. Volume-of-fluid methods, like level-set methods, do not require special procedures to model topological changes of the front. For this reason level set methods and volume-of-fluid methods are somewhat easier to implement in three dimensions than piecewise polynomial methods, while retaining the improved accuracy obtained by explicitly modeling the interface motion.

A fundamental property of all volume-of-fluid methods is that they are based on discretizing the *volume,* or equivalently the *volume fraction f,* of one of the fluids. Therefore, since in an incompressible flow conservation of mass is equivalent to conservation of volume, the motion of the interface in an incompressible flow is modeled by solving a conservation law for the volume fraction. As a consequence one can use a conservative finite difference method to update the volume fractions and, except for errors that occur as a result of numerical overshoot and undershoot, the volume (mass) of each fluid in the computation is conserved.

It is interesting to contrast this with the discretization in a level set method. In a level set method the level set function $\phi$, which is initially taken to be the (signed) distance from the interface (e.g., see [54]), also satisfies a conservative equation. However, conservation of $\phi$ does not imply that the volume of each fluid is conserved. It has been found that this can lead to loss of mass (cf. page 11 of Sussman, Smereka, and Osher [54]). Chang *et al.* [10] and Sussman *et al.* [52, 53] have since developed various "reinitialization" techniques for maintaining $\phi$ as a distance function that ameliorates this problem.

The algorithm we present here is based on a volume-of-fluid formulation. Volume-of-fluid methods have been in use for several decades. One of the early algorithms of this type was the SLIC algorithm of Noh and Woodward [36]. Chorin [12] developed an improved version of SLIC in order to model flame propagation and combustion. Ghoniem, Chorin, and Oppenheim [16] and Sethian [47] have used this idea to model turbulent combustion. Another variation of the volume-of-fluid approach was the VOF algorithm of Hirt and Nichols [25]. Several codes based on the VOF idea, namely, SOLA-VOF [25, 35] and its descendants NASA-VOF2D [57], NASA-VOF3D [58], RIPPLE [27, 28], and FLOW3D [24] are widely used for modeling interfaces and free surfaces in industrial applications. For example, a modified version of SOLA-VOF has been used by researchers at Xerox to model the flow in thermal ink jet devices [15, 56] and SOLA-VOF and FLOW3D have been used extensively by material scientists to model problems involving melting and solidification (e.g., [9, 31, 59]).

However, all of these codes are built around a relatively crude volume-of-fluid interface reconstruction algorithm that relies on a piecewise-constant or "staircase" representation of the interface and advection algorithms that are at best first-order accurate. More modern volume-of-fluid interface reconstruction methods use a linear approximation to the interface in each multi-fluid cell (e.g., see [39, 45, 63]) that results in a piecewise-linear approximation to the interface. For example, Youngs and his colleagues [63, 64] have used several first-order, piecewise-linear algorithms coupled with numerical integration schemes for gas dynamics to model turbulent mixing and the Rayleigh–Taylor and Richtmyer–Meshkov instabilities in compressible flows. Youngs [64] also applied a piecewise linear algorithm for compressible flow in three dimensions (also see [26, 45]). However, all of the above-cited piecewise linear interface reconstruction algorithms still only produce a first-order accurate approximation to the front.

In recent work Puckett [42] and Pilliod and Puckett [41] introduced two second-order volume-of-fluid interface reconstruction algorithms. One of these algorithms, the least squares volume-of-fluid interface reconstruction algorithm (LVIRA) [42]) has been coupled to a second-order Godunov algorithm for gas dynamics for modeling compressible flows with two fluids having different equations of state. This algorithm has been used extensively to compute shock refraction [22, 23, 21, 44, 43] and Richtmyer–Meshkov instability [33].

The goal of this paper is to develop a second-order algorithm for incompressible flows with two fluids of different densities. The basis for the numerical algorithm is a variable density version of the approximate projection method developed by Almgren, Bell, and Szymczak [2] for the incompressible Navier–Stokes equations. This algorithm is based on high-resolution upwind methods that provide a robust discretization of the underlying flow equations, making it a suitable framework for modeling variable density flows with a sharp interface. The density interface is constructed from volume fractions using the LVIRA reconstruction algorithm. We present both an operator-split advection algorithm and an unsplit advection algorithm due to Pilliod and Puckett [41] for advecting the volume fractions to update the density field. Both of these approaches use the geometric information from the reconstructed interface to compute fluxes in the volume fraction update, rather than a simple upwind discretization of it. Consequently, the front does not diffuse as it would if one were to use a standard conservative finite difference method to update the values of $f$. The resulting algorithm is second-order accurate for smooth flows containing a smooth interface while degrading to first-order accuracy when the velocity field or interface is not smooth.

The remainder of this paper is organized as follows. In the next two sections we give a detailed description of the interface tracking method, review the basic projection algorithm, and discuss how the VOF interface tracking algorithm is coupled to the basic flow algorithm. In Section 4 we present computational examples that demonstrate the convergence properties of the algorithm and demonstrate its performance on several realistic model problems. In Section 5 we state our conclusions.

## 2. THE INTERFACE TRACKING ALGORITHM

In this section we describe a method for tracking the interface between two materials, say a dark fluid and a light fluid, in a two-dimensional, incompressible, nonreacting flow. We consider the problem of advancing a front in a known divergence-free velocity field $\mathbf{u} = (u, v)$. In the following section we describe our method for obtaining this velocity field as a solution of the Navier–Stokes equations. We begin by covering the problem domain with a uniform grid with spacing $h = \Delta x = \Delta y$. With each grid cell we associate a number $f_{i,j}$ that represents the fraction of the $(i, j)$th cell that is occupied by dark fluid; i.e.,

$$f_{i,j}h^2 = \text{volume of dark fluid in the } (i, j)\text{th cell.} \tag{2}$$

The number $f_{i,j}$ is called the *volume fraction* (of dark fluid) in the $(i, j)$th cell. It is apparent that

$$0 \le f_{i,j} \le 1, \tag{3}$$

that the volume fraction associated with the light fluid is $(1 - f_{i,j})$, and that a portion of the interface lies in the $(i, j)$th cell if and only if $0 < f_{i,j} < 1$. The discrete variable $f_{i,j}$ is a discretization of the characteristic function associated with the dark fluid,

$$f(x, y) \equiv \begin{cases} 1 & \text{if there is dark fluid at the point } (x, y), \\ 0 & \text{if there is light fluid at the point } (x, y), \end{cases} \tag{4}$$

in the sense that

$$f_{i,j}h^2 \approx \int\int_{i,j\text{th cell}} f(x, y)\, dx\, dy. \tag{5}$$

Since the fluid type does not change along particle paths the characteristic function $f$ is passively advected with the flow. Hence $f$ satisfies the advection equation,

$$f_t + uf_x + vf_y = 0. \tag{6}$$

Since the flow is incompressible, $\mathbf{u}$ satisfies

$$u_x + v_y = 0. \tag{7}$$

Multiplying (7) by $f$ and adding it to (6) we obtain a conservation law for the characteristic function $f$,

$$f_t + (uf)_x + (vf)_y = 0. \tag{8}$$

Equation (8) reflects the fact that in an incompressible flow conservation of mass is equivalent to conservation of volume and, hence, conservation of $f$. If one uses a conservative finite difference method to update $f_{i,j}$ then the mass of dark fluid is conserved,

$$h^2 \sum_{i,j} f^0_{i,j} = \cdots = h^2 \sum_{i,j} f^n_{i,j} \cdots h^2 \sum_{i,j} f^{n+1}_{i,j}. \tag{9}$$

Here $f^n_{i,j}$ denotes the volume fraction in the $(i, j)$th cell at time $t^n = n\,\Delta t$ and we have assumed that no mass enters or leaves the problem domain during the computation.

During the numerical solution of (8) numerical overshoot and undershoot may cause some volume fractions to violate (3). Although in the computations presented here we found that this effect was negligible, we satisfy (3) explicitly by truncating the values of volume fractions which leave the interval [0, 1] during the advection step,

$$f_{i,j}^{n+1} := \min(1, \max(f_{i,j}^{n+1}, 0)), \tag{10}$$

where := denotes assignment in place. In the computations presented here we have found that the net change in total mass over the entire length of the computation is at most one hundredth of one percent of the total mass of either fluid. We note that one can also remedy overshoot and undershoot with a technique similar to the flux redistribution ideas in [5, 11] (e.g., see [46]).

## 2.1. *The Interface Reconstruction Algorithm*

In order to advance the solution of (8) in time we first need to construct an approximation to the interface given the values of the volume fractions $f_{i,j}^n$ at time $t = n\,\Delta t$. We refer to an algorithm for doing this as a volume-of-fluid *interface reconstruction algorithm.* In this work we use a volume-of-fluid interface reconstruction algorithm known as the least squares volume-of-fluid interface reconstruction algorithm (LVIRA) due to Puckett [41, 42]. This algorithm produces a linear approximation to the interface in each multifluid cell, i.e., each cell for which $0 < f_{i,j} < 1$. In general, this piecewise linear approximation is not continuous. The LVIRA algorithm uses the volume fractions $f_{i,j}$ in a $3 \times 3$ block of cells to determine the approximate interface in the center cell of the block. An important property of this algorithm, common to all volume-of-fluid algorithms, is that the approximate interface is always chosen so that it reproduces the correct (i.e., given) volume fraction in each multifluid cell. In other words, if we denote the volume fraction due to the linear approximation in the $(i, j)$th cell by $\tilde{f}_{i,j}$, then we require that

$$\tilde{f}_{i,j} = f_{i,j} \quad \text{for all } i, j. \tag{11}$$

The LVIRA algorithm also returns a vector $\mathbf{n}$ normal to the interface. In this article we adopt the convention that $\mathbf{n}$ always points away from the dark fluid. The normal vector $\mathbf{n}_{i,j}$ in the $(i, j)$th cell, together with the given volume fraction $f_{i,j}$, uniquely determines the approximate linear interface in the $(i, j)$th cell. Thus, since the volume fraction $f_{i,j}$ is given, the interface reconstruction algorithm is simply a rule for determining a unit normal vector from the $3 \times 3$ block of volume fractions centered on the $(i, j)$th cell.

In the LVIRA algorithm we choose the unit normal $\mathbf{n}$ which minimizes the function

$$G_{i,j}(\mathbf{n}) = \sum_{k,l=-1}^{1} (f_{i+k,j+l} - \tilde{f}_{i+k,j+l}(\mathbf{n}))^2, \tag{12}$$

where $f_{i+k,j+l}$ are the given volume fractions in the $3 \times 3$ block centered on the $(i, j)$th cell and $\tilde{f}_{i+k,j+l}(\mathbf{n})$ are the volume fractions due to the line with unit normal $\mathbf{n}$ which divides the $(i, j)$th cell into fractions $f_{i,j}$ and $1 - f_{i,j}$. Our approximation to the interface is this line.

This algorithm has the property that if the original volume fractions in the $3 \times 3$ block centered on the $(i, j)$th cell are due to a linear interface, then it will reproduce the interface exactly. We conjecture that the approximate interface will be a second-order approximation to the true interface whenever the true interface is $C^2$. Computational tests presented in [40, 41] support this conjecture and indicate that the algorithm is first-order accurate whenever the interface fails to be this smooth.[1]

We note that the basic principles of LVIRA are applicable to both axisymmetric and three-dimensional interface reconstruction. LVIRA will exactly reconstruct conical interfaces in axisymmetric geometry and planes in three dimensions.

## 2.2. *The Volume-of-Fluid Advection Algorithm*

The second step in the solution of (8) is an algorithm for evolving the volume fractions in time. Suppose that at time $t^n = n\,\Delta t$ we have values of the velocity field $(u_{i\pm1/2}, v_{i,j\pm1/2})$ defined at the centers of the cell edges and that these velocities satisfy a discrete form of (7),

$$\frac{(u_{i+1/2,j} - u_{i-1/2,j})}{\Delta x} + \frac{(v_{i,j+1/2} - v_{i,j-1/2})}{\Delta y} = 0. \tag{13}$$

Given an approximation to the interface in each cell for which $0 < f_{i,j}^n < 1$ we wish to determine the volume fractions $f_{i,j}^{n+1}$ at the new time $t^{n+1} = (n + 1)\,\Delta t$. We refer to algorithms for doing this as volume-of-fluid *advection algorithms.*

In our work we have used two types of advection algorithms. Both are based on the standard conservative finite difference update of (8),

$$f_{i,j}^{n+1} = f_{i,j}^n + \frac{\Delta t}{\Delta x}[F_{i-1/2,j} - F_{i+1/2,j}] + \frac{\Delta t}{\Delta y}[G_{i,j-1/2} - G_{i,j+1/2}], \tag{14}$$

where $F_{i-1/2,j} = (fu)_{i-1/2,j}$ denotes the flux of $f$ across the left edge of the $(i, j)$th cell and $G_{i,j-1/2} = (fv)_{i,j-1/2}$ denotes the flux across the bottom edge of the $(i, j)$th cell, etc.

[1] In [40, 41] it is demonstrated numerically that these algorithms are first/second-order in the following sense. If the interface between materials 1 and 2 is $C^2$ with bounded second derivative, then the $L^1$ norm of the difference between the true characteristic function for material 1 and the approximate characteristic function decays at the appropriate rate. It is also shown (numerically) in [40, 41] that when the interface fails to be $C^2$ everywhere the algorithm exhibits first-order convergence.

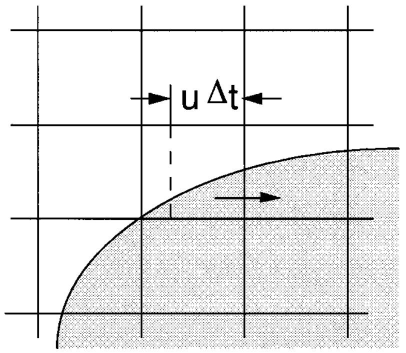

**FIG. 1.** The fraction of dark fluid to the right of the dotted line is advected into the neighboring cell on the right. In this example $u$ is positive.

2.2.1. The Operator Split Advection Algorithm. The simplest of the two advection algorithms is the *fractional step* or *operator split* method,

$$\tilde{f}_{i,j} = f^n_{i,j} + \frac{\Delta t}{\Delta x}[F_{i-1/2,j} - F_{i+1/2,j}], \tag{15}$$

$$f^{n+1}_{i,j} = \tilde{f}_{i,j} + \frac{\Delta t}{\Delta y}[\tilde{G}_{i,j-1/2} - \tilde{G}_{i,j+1/2}]. \tag{16}$$

There is a simple geometric interpretation of the fluxes in (15)–(16). Suppose that $u_{i+1/2,j}$ is positive. Divide cell $(i, j)$ into two disjoint rectangles, with areas $u_{i+1/2,j}\,\Delta t\,\Delta y$ on the right and $(\Delta x - u_{i+1/2,j}\,\Delta t)\,\Delta y$ on the left, as shown in Fig. 1. All of the fluid in the first rectangle will cross the right-hand edge during this time step. In particular, the flux of dark fluid across this edge is equal to the amount of dark fluid contained in this rectangle. In a volume-of-fluid method, this is determined by the location of the reconstructed interface. Thus, if $V_{i+1/2,j}$ denotes the volume of dark fluid in this rectangle then the flux is given by

$$F_{i+1/2,j} = u_{i+1/2,j}V_{i+1/2,j}/(u_{i+1/2,j}\,\Delta t\,\Delta y) = V_{i+1/2,j}/(\Delta t\,\Delta y). \tag{17}$$

After using (17) in (15) to determine the intermediate volume fractions $\tilde{f}_{i,j}$, one then uses these values to reconstruct the interface in all cells which satisfy $0 < \tilde{f}_{i,j} < 1$. The vertical fluxes $\tilde{G}_{i,j+1/2}$ are then determined by a geometric construction analogous to the one described for the horizontal fluxes, and the volume fractions at the new time level $f^{n+1}_{i,j}$ are found by inserting these vertical fluxes into (16). This procedure can be made second-order simply by alternating the sweep direction at each time step; i.e., by employing "Strang splitting" [51].

In our work we have modified the algorithm in (15)–(16) so that we compute a solution of (6) rather than (8) in each sweep direction. In other words, we approximate solutions of

$$f_t + (fu)_x = fu_x, \tag{18}$$

$$f_t + (fv)_y = fv_y, \tag{19}$$

rather than

$$f_t + (fu)_x = 0, \tag{20}$$

$$f_t = (fv)_y = 0. \tag{21}$$

In order to maintain conservation of $f$ it is now necessary to discretize $f$ implicitly on the right-hand side of (18) and explicitly in the right-hand side of (19),

$$\tilde{f}_{i,j} = \left\{ f^n_{i,j} + \frac{\Delta t}{\Delta x}[F_{i-1/2,j} - F_{i+1/2,j}] \right\} \Bigg/ \left[ 1 - \frac{\Delta t}{\Delta x}(u_{i+1/2,j} - u_{i-1/2,j}) \right] \tag{22}$$

$$f^{n+1}_{i,j} = \tilde{f}_{i,j}\left\{ 1 + \frac{\Delta t}{\Delta y}(v_{i,j+1/2} - v_{i,j-1/2}) \right\} + \frac{\Delta t}{\Delta y}[\tilde{G}_{i,j-1/2} - \tilde{G}_{i,j+1/2}]. \tag{23}$$

To obtain second-order accuracy we simply alternate the sweep directions at each time step, taking care that during the first sweep $f$ is differenced implicitly and that during the second sweep $f$ is differenced explicitly.

2.2.2. A Second-Order Unsplit Advection Algorithm. In many cases one will obtain satisfactory results with the second-order fractional step method described above. However, for some problems, such as unstable displacements in porous media, fractional step methods can distort the interface (e.g., see the discussion in [6] and the references therein). A characteristic feature of this problem in the so-called "push–pull" or "staircase" phenomenon. For problems such as these it is preferable to use an unsplit advection algorithm. In this work we have used an unsplit, volume-of-fluid advection algorithm due to Pilliod [40, 41] that is based on an unsplit advection algorithm for scalar hyperbolic conservation laws due to Bell, Dawson, and Shubin [6].

We wish to use the conservative finite difference formula (14) to solve the conservation law (8). To illustrate our approach we assume that $u > 0$ and $v > 0$ and describe how to determine the flux $F_{i+1/2,j}$. The other cases are

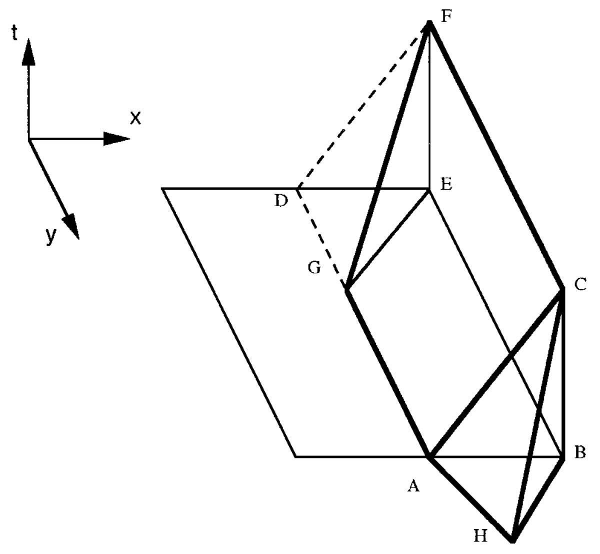

**FIG. 2.** All of the fluid in this solid will pass through the right hand edge of the cell in the interval ($t$, $t + dt$).

analogous. The flux through the right-hand edge of the ($i$, $j$)th cell in the time interval ($t^n$, $t^{n+1}$) is

$$\begin{aligned} F_{i+1/2,j} &= \int_{t^n}^{t^{n+1}} \int_{(j-1/2)h}^{(j+1/2)h} u(x, y, t) f(x_{i+1/2,j}, y, t)\, dy\, dt \\ &= u_{i+1/2,j} \int_{t^n}^{t^{n+1}} \int_{(j-1/2)h}^{(j+1/2)h} f(x_{i+1/2,j}, y, t)\, dy\, dt, \end{aligned} \tag{24}$$

where we have assumed that in our numerical discretization $u_{i+1/2,j}$ is constant on the space–time interval $(y_{i,j-1/2}, y_{i,j+1/2}) \times (t^n, t^{n+1})$. This integral is the amount of dark fluid in the space–time rectangle $BCEF$ shown in Fig. 2. We can find this volume by tracking back along the characteristics that originate from the rectangle. This gives us the solid region defined by the points $ABCEFGH$ shown in Fig. 2.

This solid is the prism $ABCDEF,$ minus the tetrahedron $DEFG,$ plus the tetrahedron $ABCH.$ We approximate the flux $F_{i+1/2,j}$ in (24) by integrating (8) over the prism and integrating by parts. Writing (8) in the form

$$f_t + u_x f + u f_x + (vf)_y = 0 \tag{25}$$

and setting $u = u_{i+1/2,j}$ and $u_x = (u_x)_{i,j}$, we find that

$$\iiint_{ABCDEF} (f_t + (u_x)_{i,j} f + u_{i+1/2,j} f_x + (vf)_y)\, dx\, dy\, dt = 0. \tag{26}$$

Integrating the above expression by parts and noting that $u_{i+1/2,j}$ is constant, we find that the flux $F_{i+1/2,j}$ is given by

$$\begin{aligned} F_{i+1/2,j} = \iint_{ABDE} f\, dx\, dy + \iint_{ABC} vf\, dx\, dt \\ - \iint_{DEF} vf\, dx\, dt + \iiint_{ABCDEF} (u_x)_{i,j} f\, dx\, dy\, dt. \end{aligned} \tag{27}$$

The integral over $ABDE$ is the volume of dark fluid in this rectangle. In order to evaluate this volume we use the LVIRA algorithm to determine an approximation to the interface in the ($i$, $j$)th cell and then compute the area of the intersection of the dark fluid with rectangle $ABDE.$ Now let $R_1$ be the ratio of the volume of dark fluid in $ABDE$ to the area of $ABDE$ and let $V_1$ be the volume of the prism $ABCDEF.$ The volume integral in (27) is approximately

$$\iiint_{ABCDEF} (u_x)_{i,j} f\, dx\, dy\, dt \approx R_1 V_1 (u_x)_{i,j}.$$

Next we explain how to evaluate the integral over $DEF.$ The domain of dependence of $DEF$ is the triangle $DEG.$ The tetrahedron $DEFG$ is related to the triangle $DEF$ through

$$\begin{aligned} \iint_{DEF} f\, dx\, dt = \iint_{DEG} f\, dx\, dy \\ + \iiint_{DEFG} ((u_x)_{i,j} + (v_y)_{i,j}) f\, dx\, dy\, dt. \end{aligned} \tag{28}$$

The integral over $DEG$ is the volume of dark fluid in the triangle $DEG.$ In order to evaluate the volume of dark fluid in $DEG$ we use the LVIRA algorithm to determine an approximation to the interface in the ($i$, $j$)th cell and then compute the area of the intersection of the dark fluid with triangle $DEG.$

Let $R_2$ be the ratio of the volume of dark fluid in $DEG$ to the area of $DEG,$ and let $V_2$ be the volume of the tetrahedron $DEFG.$ Then the volume integral in (28) is approximately

$$\iiint_{DEFG} ((u_x)_{i,j} + (v_y)_{i,j}) f\, dx\, dy\, dt = R_2 V_2 ((u_x)_{i,j} + (v_y)_{i,j}),$$

which, by (13), is approximately zero. The integral over $ABC$ is performed similarly. Thus we are able to evaluate each term of (27) and, hence, determine the flux of dark fluid through the right edge of the cell.

Note that if $v_{i,j+1/2} < 0$, then the point $G$ will lie in the ($i$, $j$ + 1)th cell. Thus the tetrahedron $DEFG$ will lie in the ($i$, $j$ + 1)th cell instead of the ($i$, $j$)th cell. In this case we add the tetrahedron $DEFG$ to the prism $ABCDEF,$ instead of substracting it as we did above. Thus in (27) we add the integral over $DEF$ instead of subtracting it.

In order to avoid the distorted region that arises when

$u_{i+1/2,j}$ and $u_{i+1/2,j+1}$ are of opposite sign we determine the $x$-coordinate of the vertex $G$ by

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

Thus we are assured that if $v_{i,j+1/2} < 0$ then $G$ lies in the $(i, j+1)$th cell. In addition to simplifying the procedure, the presence of shear indicated by the sign change in the velocity makes the implicit assumption of a zero velocity at the cell corner reasonable.

## 3. THE VARIABLE DENSITY PROJECTION METHOD

### 3.1. *The Equations of Motion*

We solve the variable density Navier–Stokes equations subject to the incompressibility constraint, together with advection equations for density and viscosity.

$$U_t + (U\cdot\nabla)U = \frac{1}{\rho}(\nabla\cdot(2\mu\mathscr{D}) - \nabla p + \mathbf{F}), \tag{29}$$

$$\nabla\cdot U = 0, \tag{30}$$

$$\rho_t + (U\cdot\nabla)\rho = 0, \tag{31}$$

$$\mu_t + (U\cdot\nabla)\mu = 0. \tag{32}$$

Here $U = (u(x, y, t), v(x, y, t))$ is the velocity, $p = p(x, y, t)$ is the hydrodynamic pressure, $\rho = \rho(x, y, t)$ is the density, $\mu = \mu(x, y, t)$ is the dynamic viscosity, $\mathbf{F}$ represents body forces such as gravity, and $\mathscr{D}$ is the viscous stress tensor, $\mathscr{D} = 1/2(\nabla U + \nabla U^{\mathrm{T}})$.

We describe the algorithm for approximating solutions of (29)–(32) with Dirichlet boundary conditions on velocity and uniform grid spacing, $\Delta x = \Delta y = h$. Our strategy for solving the above system of equations is a fractional step scheme having four parts: solving the momentum equation (29) for velocity using a lagged pressure gradient, advancing the volume fractions via the procedure described in Section 2, using this information to reconstruct the density and dynamic viscosity by volume weighting

$$\rho_{i,j} = f_{i,j}\rho_1 + (1 - f_{i,j})\rho_2, \tag{33}$$

$$\mu_{i,j} = f_{i,j}\mu_1 + (1 - f_{i,j})\mu_2, \tag{34}$$

and projecting the intermediate velocity field onto the space of approximately discretely divergence-free vector fields. Here $\rho_1$ and $\rho_2$ are the densities of the ‘‘dark’’ and ‘‘light’’ fluids, respectively.

In the ‘‘advection–diffusion step’’, we solve

$$\frac{U^* - U^n}{\Delta t} = -[(U\cdot\nabla)U]^{n+1/2} + \frac{1}{\rho^{n+1/2}}((\mathscr{L}_h^* + \mathscr{L}_h^n) - \nabla p^{n-1/2} + \mathbf{F}^{n+1/2}) \tag{35}$$

for the intermediate velocity $U^*$, where $\mathscr{L}_h$ is a second-order finite difference approximation to $\nabla\cdot(\mu\mathscr{D})$. The nonlinear advection terms $[(U\cdot\nabla)U]^{n+1/2}$, are evaluated using an explicit predictor–corrector scheme and require only the available data at $t^n$. The density, $\rho$, and viscosity, $\mu$, are constructed from the volume fractions calculated at time $t^{n+1/2}$ in the interface tracking step using the predicted velocities at cell edges obtained in the process of differencing $[(U\cdot\nabla)U]^{n+1/2}$. The lagged pressure gradient $\nabla p^{n-1/2}$, and force, $\mathbf{F}^{n+1/2}$, are treated as source terms. Because the viscosity coefficient, $\mu$, is a function of volume fraction and therefore varies in space, the implicit part of (35) corresponds to a coupled parabolic solve for both velocity components.

The velocity field $U^*$ is not, in general, divergence-free. The projection step of the algorithm decomposes the result of the first step into a scalar multiple of a discrete gradient of a scalar potential (the scalar multiple being the inverse density) and an approximately discretely divergence-free vector field, which correspond, respectively, to an update to the pressure gradient term and an update to the velocity. In particular, if $\mathbf{P}$ represents the approximate projection operator, then

$$\frac{U^{n+1} - U^n}{\Delta t} = \mathbf{P}\left(\frac{U^* - U^n}{\Delta t}\right), \tag{36}$$

$$\frac{1}{\rho^{n+1/2}}\nabla p^{n+1/2} = \frac{1}{\rho^{n+1/2}}\nabla p^{n-1/2} + (\mathbf{I} - \mathbf{P})\left(\frac{U^* - U^n}{\Delta t}\right). \tag{37}$$

*Discretization of the Advection Terms.* The discretization of the advection terms in this algorithm is a variable density version of the method discussed by Bell, Colella, and Howell [4]. It is a predictor–corrector method based on the unsplit Godunov method introduced by Colella [14]. In the predictor, velocities are extrapolated along characteristics to the cell edges at the new half-time-step level, $t^{n+1/2}$. A MAC projection (e.g., see [4]) is applied to make these edge velocities divergence-free. The volume fractions and velocities are advanced using these edge velocities.

In the predictor we extrapolate the velocity to the cell edges at $t^{n+1/2}$ using a second-order Taylor series expansion in space and time. The time derivative is replaced using (29). For edge $(i + \frac{1}{2}, j)$ this gives

$$U_{i+1/2,j}^{n+1/2,L} = U_{ij}^n + \left(\frac{\Delta x}{2} - \frac{u_{ij}\,\Delta t}{2}\right)U_{x,ij}^n - \frac{\Delta t}{2}\widehat{(vU_y)}_{ij} + \frac{1}{\rho_{ij}^n}\frac{\Delta t}{2}(\mathscr{L}_{ij}^n - Gp_{ij}^{n-1/2} + \mathbf{F}_{ij}^n) \tag{38}$$

extrapolated from cell $(i, j)$, and

$$U^{n+1/2,R}_{i+1/2,j} = U^n_{i+1,j} - \left(\frac{\Delta x}{2} + \frac{u_{i+1,j}\,\Delta t}{2}\right) U^n_{x,i+1,j} - \frac{\Delta t}{2}\,(\widehat{vU_y})_{i+1,j} + \frac{1}{\rho^n_{i+1,j}}\frac{\Delta t}{2}\,(\mathscr{L}^n_{i+1,j} - Gp^{n-1/2}_{i+1,j} + \mathbf{F}^n_{i+1,j}) \tag{39}$$

extrapolated from cell $(i + 1, j)$. Here and in what follows, $D$ and $G$ represent discrete approximations to the divergence and gradient operators, respectively.

Analogous formulae are used to predict values at each of the other edges of the cell. In evaluating these terms the first-order derivatives normal to the edge (in this case $U_x$) are evaluated using a monotonicity-limited fourth-order slope approximation [13]. The limiting is done on the components of the velocity individually.

The transverse derivative terms ($\widehat{vU_y}$ above) are evaluated as in [4], by first extrapolating from above and below to construct edge states, using normal derivatives only, and then choosing between these states using the upwinding procedure defined below. In particular, we define

$$\hat{U}^B_{i,j+1/2} = U^n_{ij} + \left(\frac{\Delta y}{2} - \frac{v_{ij}\,\Delta t}{2}\right) U^n_{y,ij},$$

$$\hat{U}^T_{i,j+1/2} = U^n_{i,j+1} - \left(\frac{\Delta y}{2} + \frac{v_{i,j+1}\,\Delta t}{2}\right) U^n_{y,i,j+1},$$

where $U_y$ are limited slopes in the $y$ direction, with similar formulae for the lower edge of cell $B_{i,j}$. In this upwinding procedure we first define the normal velocity on the edge:

$$\hat{v}^{\text{adv}}_{i,j+1/2} = \begin{cases} \hat{v}^B & \text{if } \hat{v}^B > 0, \hat{v}^B + \hat{v}^T > 0, \\ 0 & \text{if } \hat{v}^B \le 0, \hat{v}^T \ge 0 \text{ or } \hat{v}^B + \hat{v}^T = 0, \\ \hat{v}^T & \text{if } \hat{v}^T < 0, \hat{v}^B + \hat{v}^T < 0. \end{cases}$$

(Here and in the next equation we suppress the $i$, $j$ + 1/2 spatial indices on bottom and top states.) We now upwind $\hat{U}$ based on $\hat{v}^{\text{adv}}_{i,j+1/2}$:

$$\hat{U}_{i,j+1/2} = \begin{cases} \hat{U}^B & \text{if } \hat{v}^{\text{adv}}_{i,j+1/2} > 0, \\ 1/2(\hat{U}^B + \hat{U}^T) & \text{if } \hat{v}^{\text{adv}}_{i,j+1/2} = 0, \\ \hat{U}^T & \text{if } \hat{v}^{\text{adv}}_{i,j+1/2} < 0. \end{cases}$$

After constructing $\hat{U}_{i,j-1/2}$ in a similar manner, we use these upwind values to form the transverse derivative in (38),

$$(\widehat{vU_y})_{ij} = \frac{1}{2\,\Delta y}(\hat{v}^{\text{adv}}_{i,j+1/2} + \hat{v}^{\text{adv}}_{i,j-1/2})(\hat{U}_{i,j+1/2} - \hat{U}_{i,j-1/2}).$$

We use a similar upwinding procedure to choose the appropriate states $U_{i+1/2,j}$ given the left and right states, $U^{n+1/2,\text{L}}_{i+1/2,j}$ and $U^{n+1/2,\text{R}}_{i+1/2,j}$:

$$U_{i+1/2,j} = \begin{cases} U^{\text{L}} & \text{if } u^{\text{L}} > 0 \text{ and } u^{\text{L}} + u^{\text{R}} > 0, \\ 1/2(U^{\text{L}} + U^{\text{R}}) & \text{if } u^{\text{L}} \le 0, u^{\text{R}} \ge 0, \text{ or } u^{\text{L}} + u^{\text{R}} = 0, \\ U^{\text{R}} & \text{if } u^{\text{R}} < 0 \text{ and } u^{\text{L}} + u^{\text{R}} < 0, \end{cases}$$

where we have again suppressed the spatial indices $i + \frac{1}{2}$, $j$. We follow a similar procedure to construct $U_{i-1/2,j}$, $U_{i,j+1/2}$, and $U_{i,j-1/2}$.

In general the normal velocities at the edges are not divergence-free; in order to make these velocities divergence-free we apply the MAC projection [4] before construction of the convective derivatives. The equation

$$D^{\text{MAC}}\left(\frac{1}{\rho^n}G^{\text{MAC}}\phi\right) = D^{\text{MAC}}U^{n+1/2}$$

is solved for $\phi$, where

$$D^{\text{MAC}}U^{n+1/2} = \left(\frac{u^{n+1/2}_{i+1/2,j} - u^{n+1/2}_{i-1/2,j}}{\Delta x} + \frac{v^{n+1/2}_{i,j+1/2} - v^{n+1/2}_{i,j-1/2}}{\Delta y}\right)$$

and

$$(G^{\text{MAC}}\phi)^x_{i+1/2,j} = \frac{\phi_{i+1,j} - \phi_{i,j}}{\Delta x},$$

$$(G^{\text{MAC}}\phi)^y_{i,j+1/2} = \frac{\phi_{i,j+1} - \phi_{i,j}}{\Delta y}.$$

We then define advection velocities by

$$u^{\text{ADV}}_{i+1/2,j} := u^{n+1/2}_{i+1/2,j} - \frac{1}{\rho^n_{i+1/2,j}}(G^{\text{MAC}}\phi)^x_{i+1/2,j},$$

$$v^{\text{ADV}}_{i,j+1/2} := v^{n+1/2}_{i,j+1/2} - \frac{1}{\rho^n_{i,j+1/2}}(G^{\text{MAC}}\phi)^y_{i,j+1/2},$$

where $\rho_{i+1/2,j}$ and $\rho_{i,j+1/2}$ are defined by

$$\frac{1}{\rho_{i+1/2,j}} \equiv \frac{1}{2}\left(\frac{1}{\rho_{i,j}} + \frac{1}{\rho_{i+1,j}}\right),$$

$$\frac{1}{\rho_{i,j+1/2}} \equiv \frac{1}{2}\left(\frac{1}{\rho_{i,j}} + \frac{1}{\rho_{i,j+1}}\right).$$

In the corrector step, we form an approximation to the convective derivatives in (29)

$$[(U\cdot\nabla)U]^{n+1/2} = \frac{1}{2\,\Delta x}(u^{\mathrm{ADV}}_{i+1/2,j} + u^{\mathrm{ADV}}_{i-1/2,j})(U^{n+1/2}_{i+1/2,j} - U^{n+1/2}_{i-1/2,j}) + \frac{1}{2\,\Delta y}(v^{\mathrm{ADV}}_{i,j+1/2} + v^{\mathrm{ADV}}_{i,j-1/2})(U^{n+1/2}_{i,j+1/2} - U^{n+1/2}_{i,j-1/2}).$$

*The CFL Constraint.* In this method there are two sources of a CFL-like constraint: the volume-of-fluid advection algorithm and the differencing of the nonlinear advection terms in (29). It is apparent from geometric considerations that one must choose the CFL number $\sigma$ in the volume-of-fluid advection step so that the amount of fluid which leaves a cell in one time step is no more than the amount of fluid that was originally in the cell. In other words, one must choose $\sigma$ so that

$$V_{i+1/2,j} - V_{i-1/2,j} \le f_{i,j}\,\Delta x\,\Delta y \tag{40}$$

for all $i, j$, where $V_{i+1/2,j}$ is the volume of dark fluid which crosses the right-hand edge of the $(i, j)$th cell in one time step. One way to ensure that (40) is always satisfied is to choose $\sigma \in (0, 1]$ so that

$$|u^n_{i+1/2,j}|\,\Delta t \le \Delta x/2, \quad |v^n_{i,j+1/2}|\,\Delta t \le \Delta y/2 \quad \text{for all } i, j. \tag{41}$$

Our method of differencing the nonlinear advection terms in (29) is an explicit difference scheme and therefore also requires a time-step restriction; however, this condition is less restrictive than (41) so we use that condition to set the time step.

## 3.2. *The Approximate Projection*

Since the velocity is defined at cell centers, the projection used to enforce incompressibility at time $t^{n+1}$ must include a divergence operator that acts on cell-centered quantities, unlike the MAC projection. The projection we use here is approximate in the sense that $\mathbf{P^2} \neq \mathbf{P}$. This approximate projection uses a standard finite element basis for representing the pressure so that the linear system that is solved during the projection step corresponds to the system that is solved using bilinear finite element methods for Poisson problems. The introduction of this approach to the projection, as well as the motivation for using an approximate, rather than exact, projection, is described in detail in [2] for the constant density case; here we extend the discussion to include variable density.

We consider the scalar pressure field to be a $C^0$ function that is bilinear over each cell; i.e., the pressure is in $S^h = M^1_0(x) \otimes M^1_0(y)$, where $M^t_s(x)$ is the space of polynomials of degree $t$ in the $x$ direction on each cell with $C^s$ continuity at $x$-edges. For the velocity space we define

$$\mathbf{V}^h = \mathbf{V}^{h,x} \times \mathbf{V}^{h,y},$$

where $\mathbf{V}^{h,x} = M^0_{-1}(x) \otimes M^1_{-1}(y)$ and $\mathbf{V}^{h,y} = M^1_{-1}(x) \otimes M^0_{-1}(y)$; i.e., $u$ is piecewise constant in $x$ and a discontinuous linear function of $y$ in each cell, with a similar form for $v$.

For use in the predictor and corrector, the velocity and pressure gradients are considered to be average values over each cell. The vector space, $\mathbf{V}^h$, contains additional functions that represent the linear variation within each cell. These additional degrees of freedom make $\mathbf{V}^h$ large enough to contain $\nabla\phi$ for $\phi \in S^h$. We establish a correspondence between these two representations by introducing an orthogonal decomposition of $\mathbf{V}^h$. In particular, for each vector field $V \in \mathbf{V}^h$ we define a piecewise constant component $\overline{V}_{ij}$ and the variation $V^\perp = V - \overline{V}$ so that for each cell $B_{i,j}$, $\int_{B_{i,j}} V^\perp\,d\mathbf{x} = 0$. By construction these two components are orthogonal in $L^2$ so they can be used to define a decomposition of $\mathbf{V}^h$ into two components,

$$\mathbf{V}^h = \overline{\mathbf{V}}^h \otimes \mathbf{V}^{h^\perp},$$

where $\overline{\mathbf{V}}^h$ and $\mathbf{V}^{h^\perp}$ represent the cell averages and the orthogonal linear variations, respectively. The decomposition of $\mathbf{V}^h$ induces a decomposition of $\nabla\phi$ for all $\phi \in S^h$; namely,

$$(\nabla\phi)_{ij} = (\overline{\nabla\phi})_{ij} + (\nabla\phi)^\perp_{ij}.$$

We now define a weak form of the projection on $\mathbf{V}^h$, based on a weak divergence on $\mathbf{V}^h$. In particular, we define a vector field $V^d$ in $\mathbf{V}^h$ to be divergence-free in the domain $\Omega$ if

$$\int_\Omega V^d \cdot \nabla\psi\,d\mathbf{x} = 0 \quad \forall\psi \in S^h.$$

With this definition we can then project any vector field $V$ into an inverse-density-weighted gradient $(1/\rho)\nabla\phi$ and weakly divergence-free field $V^d$ (with vanishing normal velocities on boundaries) by solving

$$\int_\Omega \frac{1}{\rho}\nabla\phi(\mathbf{x})\cdot\nabla\psi_{i+1/2,j+1/2}(\mathbf{x})\,d\mathbf{x} = \int_\Omega V\cdot\nabla\psi_{i+1/2,j+1/2}(\mathbf{x})\,d\mathbf{x} \quad \forall\psi_{i+1/2,j+1/2}(\mathbf{x}) \tag{42}$$

for $\phi(\mathbf{x}) = \sum \phi_{i+1/2,j+1/2}\psi_{i+1/2,j+1/2}(\mathbf{x})$ and by setting $V^d = \mathbf{V} - (1/\rho)\nabla\phi$. Here the $\psi$'s are the standard basis functions for $S^h$; namely $\psi_{i+1/2,j+1/2}(\mathbf{x})$ is the piecewise bilinear function having node values

$$\psi_{i+1/2,j+1/2}(\mathbf{x}_{k+1/2,\ell+1/2}) = S_{ik}S_{jl}.$$

The density $\rho$ is considered to be constant over each cell $B_{i,j}$.

For the purposes of the fractional step scheme we define $\overline{V}_{i,j}$ as the approximation to $(U^* - U^n)/\Delta t$ in (35) and

$$\overline{V}^d_{i,j} = \overline{V}_{i,j} - \frac{1}{\Lambda_{i,j}}\int_{B_{i,j}} \frac{1}{\rho_{i,j}} (\nabla\phi)_{i,j}\, dx\, dy = \overline{V}_{i,j} - \frac{1}{\rho_{i,j}} (\overline{\nabla\phi})_{i,j}$$

as the approximation to $(U^{n+1} - U^n)/\Delta t$ in (36). Here $\Lambda_{i,j}$ denotes the volume of $B_{i,j}$ and $\phi_{i,j}$ is the approximation to the update for $p$; i.e., $\phi(\mathbf{x}) = \delta p^n \equiv p^{n+1/2} - p^{n-1/2}$.

The left-hand-side of Eq. (42) is, in discrete form, a 9-point stencil approximating the Laplacian of $\phi$,

$$\begin{aligned}\left(D\frac{1}{\rho}G\phi\right)_{i+1/2,j+1/2} = \frac{1}{6h^2}\Bigg(&\frac{1}{\rho_{i,j}}(2\phi_{i-1/2,j-1/2} + \phi_{i+1/2,j-1/2}\\ &+ \phi_{i-1/2,j+1/2} - 4\phi_{i+1/2,j+1/2})\\ &+ \frac{1}{\rho_{i,j+1}}(2\phi_{i-1/2,j+3/2} + \phi_{i+1/2,j+3/2}\\ &+ \phi_{i-1/2,j+1/2} - 4\phi_{i+1/2,j+1/2})\\ &+ \frac{1}{\rho_{i+1,j}}(2\phi_{i+3/2,j-1/2} + \phi_{i+1/2,j-1/2}\\ &+ \phi_{i+3/2,j+1/2} - 4\phi_{i+1/2,j+1/2})\\ &+ \frac{1}{\rho_{i+1,j+1}}(2\phi_{i+3/2,j+3/2} + \phi_{i+1/2,j+3/2}\\ &+ \phi_{i+3/2,j+1/2} - 4\phi_{i+1/2,j+1/2})\Bigg),\end{aligned}$$

and the right-hand-side for $V = \overline{V}$ is a standard four-point divergence stencil,

$$\begin{aligned}(D\overline{V})_{i+1/2,j+1/2} = &\frac{V^x_{i+1,j} + V^x_{i+1,j+1} - V^x_{i,j} - V^x_{i,j+1}}{2\,\Delta x}\\ &+ \frac{V^y_{i,j+1} + V^y_{i+1,j+1} - V^y_{i,j} - V^y_{i+1,j}}{2\,\Delta y},\end{aligned}$$

where $V^x$ and $V^y$ are the $x$- and $y$-components of $V$, respectively.

The resulting linear system can be solved using standard multigrid methods (see [8]). In particular, we have used the standard V-cycle with Gauss–Seidel relaxation. In the presence of large density jumps, however, the standard multigrid solver can require an unreasonably large number of iterations to reach convergence (see [1], e.g., for an explanation). For these cases we use a multigrid preconditioned conjugate gradient solver with two Jacobi relaxations per level in the V-cycle preconditioner [55]. In the Rayleigh–Taylor computations shown below we observed typical improvements of 100 cycles reduced to 15. Although there is more work per iteration in the multigrid preconditioned conjugate gradient method than in the standard multigrid method, the improvement in CPU time was still significant.

## 4. COMPUTATIONAL RESULTS

In this section we demonstrate the convergence properties of both the operator-split and unsplit versions of the method and apply the unsplit version to several realistic problems. We do note that for these types of unstable interface problems the numerics are sensitive to the time step control, particularly on coarse grids where the viscous length scales are not fully resolved. In particular, for the more complex examples we have had to reduce the CFL parameter $\sigma$ to 0.1 to obtain satisfactory results. For the types of unstable interface problems being considered this more restrictive time step appears to be necessary to prevent growth of parasitic numerical modes.

### 4.1. *Rayleigh–Taylor Computations—Short Time*

The first numerical examples we compute are Rayleigh–Taylor instabilities with viscosity in order to demonstrate numerical convergence of the method. For the first case we consider the example treated by Bell and Marcus [7]. In this problem a heavy fluid lies above a light fluid in a rectangle 1 m wide by 4 m tall. The densities of the two fluids are 1.225 kg/m$^3$ and 0.1694 kg/m$^3$ while the dynamic viscosity of each fluid is $3.1304952 \times 10^{-3}$ kg/m s. The interface is initially a sine wave with amplitude 5 cm. For this case since the viscosities are equal the velocities are expected to be continuously differentiable but not $C^2$. We compute to time 0.25 using a fixed time step that satisfies the stability requirement on three different grids, $32 \times 128$, $64 \times 256$, and $128 \times 512$. By comparing the difference in the solutions on grids of adjacent resolution, averaging the fine grid solution onto the coarse one, we approximate the $L^1$ error on the coarser grid. We then use these estimates of the error to compute a numerical convergence rate. For this problem, for volume fraction we obtain a convergence rate of 1.81 for the split volume advection algorithm and 1.77 for the unsplit algorithm. For the velocity field we obtain a convergence rate of 1.72 for the split algorithm and 1.76 for the unsplit algorithm. These rates indicate that the method is performing at or near its formal second-order accuracy.

Of course, in most problems of real interest the viscosities of the two fluids are different. In this case there is an

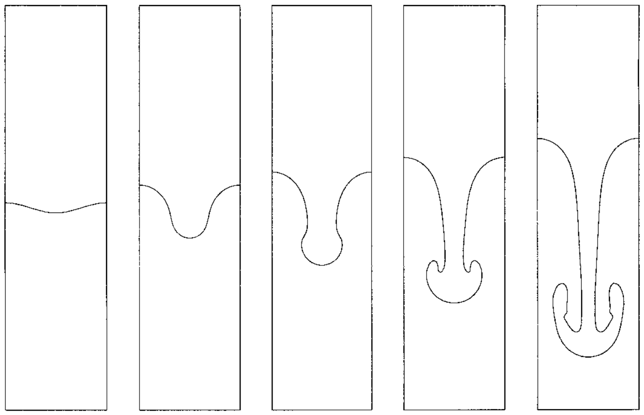

**FIG. 3.** Rayleigh–Taylor calculation with air–helium at times $t$ = 0.s, 0.047s, 0.066s, 0.088s, 0.118s.

additional loss of regularity in the solution. In particular, the velocity at the interface is not continuously differentiable; there is a jump in the velocity gradient. To test the convergence in this more realistic regime we consider an air/helium Rayleigh–Taylor instability in a 0.01 m $\times$ 0.04 m domain computing to time 0.025. For this case the viscosities of air and helium are $1.77625 \times 10^{-5}$ kg/m s and $1.941 \times 10^{-5}$ kg/m s, respectively. Using the same technique for computing convergence rates as in the previous example we obtain rates of 1.22 and 1.19 for the volume fractions for the split and unsplit algorithms, respectively. For velocity we obtain a rate of 0.91 for the split algorithm and 0.87 for the unsplit algorithm. As expected the convergence rates are reduced because of the loss of regularity of the solution; however, the method still performs at near first-order accuracy. In order to achieve second-order accuracy it would be necessary to develop a substantially more sophisticated treatment of the velocity discretization in the neighborhood of the interface; however, that is beyond the scope of this paper.

## 4.2. *A Rayleigh–Taylor Computation—Long Time*

For the next example we examine a longer time integration of the air/helium Rayleigh–Taylor instability problem from the second convergence example. For this example we have selected a relatively coarse grid of 64 $\times$ 256 to illustrate the behavior of the methodology. In Fig. 3 we show a time history of the evolution of the interface. We note that what is being plotted here and in Figs. 4 and 5 is the actual reconstructed interface, not a contour of volume fraction. We have reconstructed the interface for the purpose of plotting it using exactly the same interface reconstruction algorithm as we use in the update of the volume fractions at each time step. In particular, note that the reconstructed interface retains all of the small-scale features of the flow, and has not been smoothed as is inherent in plotting contours. In Fig. 3 one can see the development of the expected mushroom cap, even on this relatively coarse grid. In this computation the total change in mass during the entire computation was $3.04 \times 10^{-2}$ % or approximately three parts in 10,000.

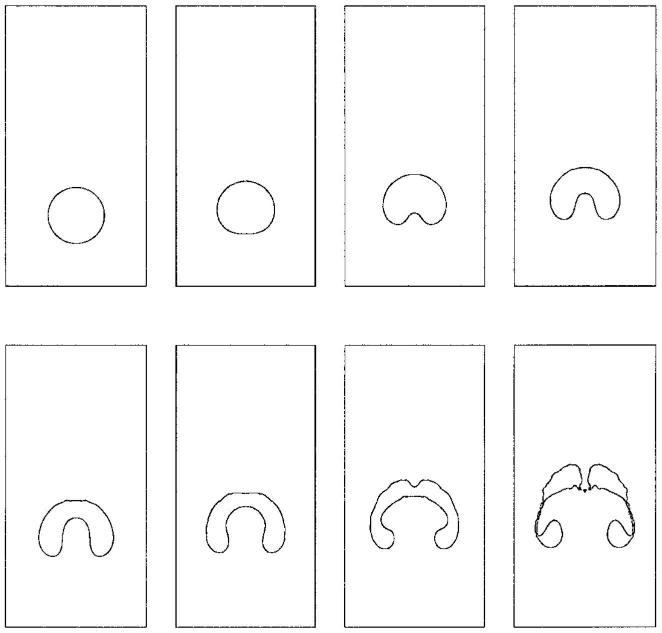

**FIG. 4.** Air bubble rising through water at times $t$ = 0.0s, 0.0101s, 0,0163s, 0,0221s, 0.0286s, 0,0357s, 0.0464s, 0.0591s.

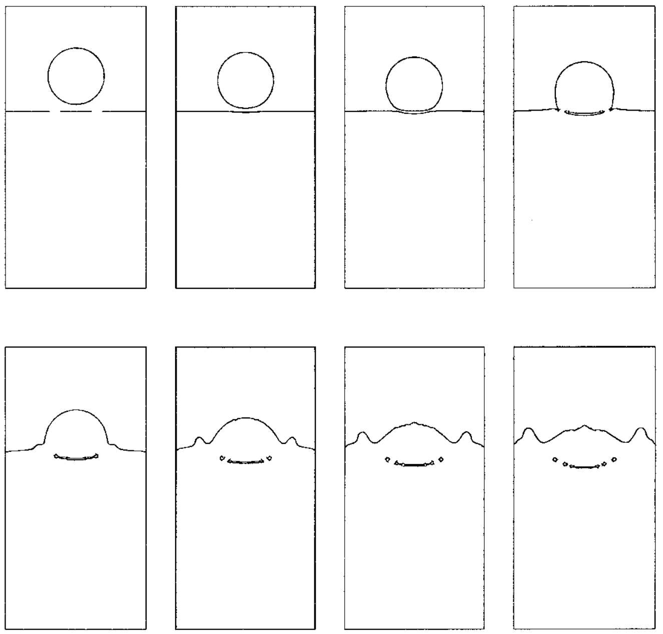

**FIG. 5.** Water droplet falling through air onto water surface at times $t$ = 0.0s, 0.00677s, 0.00980s, 0.01220s, 0.01485s, 0.01781s, 0.01995s, 0.02146s.

### 4.3. *Bubble*

For this example we consider the dynamics of a two-dimensional bubble of air in water. The viscosity of air is $1.77625 \times 10^{-5}$ kg/m s and the density is 1.225 kg/m$^3$. For water the viscosity is $1.1377 \times 10^{-3}$ kg/m s and the density is 999.2 kg/m$^3$. The computational domain is 0.007 m $\times$ 0.014 m and we have again, for purposes of illustration, used a relatively coarse grid of 64 $\times$ 128. In Fig. 4, we show a time sequence of the reconstructed interface. We note that the reconstructed interface retains small scale features of the flow. Contours of volume fraction, which are not shown, provide a considerably smoother representation of the interface. The lower edge of the bubble accelerates more quickly than the upper edge and the bubble begins to form a torus. As the bubble continues to rise the center of the bubble is stretched until the bubble breaks into two pieces. The sides of the bubble are also highly stretched and are beginning to fragment. We emphasize that these are two-dimensional bubbles in a confined box—similar to the bubbles studied experimentally by Walters and Davidson [62] although without surface tension—so the dynamics are different than radially symmetric bubbles in free space. This problem illustrates the ability of the method to handle large density and viscosity ratios and changes in topology of the interface. In this case the changes in topology occur from regions of fluid being highly stretched. In this computation the total change in mass over the length of the computation was $6.84 \times 10^{-3}$ % or approximately seven parts in 100,000.

### 4.4. *Droplet Splash*

Our final example is another air/water computation with the same computational domain and resolution as the previous example. In this case, the initial conditions are a two-dimensional water droplet above a horizontal air/water interface. This example illustrates a different type of change in topology, namely, the merging of two independent regions. Note that because the droplet is fairly close to the water surface some air bubbles are trapped in the water after the splash. Another feature of the flow is the formation of water waves resulting from the splash moving toward the walls. As before, in spite of the coarse grid the method predicts a smooth interface and handles the change in topology without any difficulty. In this computation the total change in mass over the length of the computation was $1.09 \times 10^{-4}$ % or approximately one part in 1,000,000.

## 5. CONCLUSIONS

We have presented a numerical method for approximating solutions of the incompressible Euler and Navier–Stokes equations for two fluids with different densities and viscosities. This method is based on a variable density version of the "approximate" projection method of Almgren *et al.* [2]. It incorporates the second-order volume-of-fluid interface tracking algorithms of Pilliod and Puckett [41] including an unsplit volume-of-fluid advection algorithm based on the work of Bell, Dawson, and Shubin [6].

We have demonstrated that the interface tracking algorithm is nondiffusive and that the interface is maintained as a sharp discontinuity at all times. We have also demonstrated that our method is capable of accurately computing several classes of problems that are traditionally regarded as difficult to compute, namely problems with unstable interfaces, problems with large density jumps across the interface, and problems with large viscosity jumps across the interface. Furthermore, we have demonstrated that our method is second-order accurate when the underlying flowfield is smooth and that this reduces to first-order when the flow loses regularity, such as when the velocity field loses smoothness due to a jump in viscosity at the interface. Thus, the method is performing at or near its design specification of second-order accuracy in smooth flow while reducing to first-order when the flow (or the interface) loses regularity.

In addition we have shown that for these problems the mass of each fluid is conserved to an accuracy of several parts in 10,000 to one part in a million over the length of the computation, even when the interface undergoes severe topological changes. This degree of mass conservation is an intrinsic feature of the volume-of-fluid method and is due to the fact that in a volume-of-fluid method the location of the front is updated in time by solving a conservation law for volume (8). The error in computing the total mass arises from the need to satisfy an additional positivity constraint (3). We emphasize that no additional computational procedures are required in order to attain this degree of conservation.

We have also shown that our volume-of-fluid interface reconstruction algorithm ameliorates or entirely eliminates the production of spurious bits of "flotsam" that has been observed with low order, piecewise constant reconstruction algorithms (e.g., [29, Fig. 6]).

## REFERENCES

1. R. E. Alcouffe, A. Brandt, J. E. Dendy, and J. W. Painter, The multigrid method for the diffusion equation with strongly discontinuous coefficients, *SIAM J. Sci. Stat. Comput.* **2**(4), 430 (1981).
2. A. S. Almgren, J. B. Bell, and W. G. Szymczak, A numerical method for the incompressible Navier–Stokes equations based on an approximate projection, *SIAM J. Sci. Comput.* **17**(2), (1996).
3. G. R. Baker, D. I. Meiron, and S. A. Orszag, Boundary integral methods for axisymmetric and three-dimensional Rayleigh–Taylor instability, *Physica D* **12,** 19 (1984).
4. J. B. Bell, P. Colella, and L. H. Howell, "A Multilevel Adaptive Projection Method for Unsteady Incompressible Flow," in *Proceedings, 10th AIAA Computational Fluid Dynamics Conference, Honolulu, Hawaii, June* 24–27, 1991.

5. J. B. Bell, P. Colella, and M. L. Welcome, "Conservative Front Tracking for Inviscid Compressible Flow," in *Proceedings, 22nd Annual Fluid Dynamics, Plasma Dynamics, and Lasers Conference, Honolulu, HI, 1991,* pp. 24–26.
6. J. B. Bell, C. N. Dawson, and G. R. Shubin, An unsplit, higher order godunov method for scalar conservation laws in multiple dimensions, *J. Comput. Phys.* **74,** 1 (1988).
7. J. B. Bell and D. L. Marcus, A second-order projection method for variable density flows, *J. Comput. Phys.* **101,** 334 (1992).
8. W. L. Briggs, *A Multigrid Tutorial* (SIAM, Philadelphia, 1987).
9. C. Chan, J. Mazumder, and M. M. Chen, A two-dimensional transient model for convection in a laser melted pool, *Metallurg. Trans. A* **15,** 2175 (1984).
10. Y. C. Chang, T. Y. Hou, B. Merriman, and S. Osher, A level set formulation of Eulerian interface capturing methods for incompressible fluid flows, *J. Comput. Phys.* **124,** 449 (1996).
11. I. L. Chern and P. Colella, "A Conservative Front Tracking Method for Hyperbolic Conservation Laws," Technical Report UCRL-97200, Lawrence Livermore National Laboratory, July 1987 (unpublished).
12. A. J. Chorin, Flame advection and propagation algorithms, *J. Comput. Phys.* **35,** 1 (1980).
13. P. Colella, A direct Eulerian MUSCL scheme for gas dynamics, *SIAM J. Sci. Stat. Comput.* **6,** 104 (1985).
14. P. Colella, Multidimensional upwind methods for hyperbolic conservation laws, *J. Comput. Phys.* **87,** 71 (1990).
15. N. V. Deshpande, "Fluid Mechanics of Bubble Growth and Collapse in a Therml Ink-Jet Printhead," in *SPSE/SPIES Electronic Imaging Devices and Systems Symposium, January 1989.*
16. A. F. Ghoniem, A. J. Chorin, and A. K. Oppenheim, Numerical modeling of turbulent flow in a combustion tunnel, *Phil. Trans. R. Soc. London A* **304,** 303 (1982).
17. J. Glimm and O. McBryan, A computational model for interfaces, *Adv. Appl. Math.* **6,** 422 (1985).
18. J. W. Grove, The interaction of shocks with fluid interfaces, *Adv. Appl. Math.* **10,** 201 (1989).
19. H. Haj-Hariri, Q. Shi, and A. Borhan, Effect of local property smearing on global variables, implications for numerical simulation of multiphase flows, *Phys. Fluids A* **6,** 2555 (1994).
20. J. J. Helmsen, "A Comparison of Three-Dimensional Photolithography Simulators," Ph.D. thesis, U. C. Berkeley, 1994 (unpublished).
21. L. F. Henderson and E. G. Puckett, Anomalous refraction of shock waves in materials with general equations of state. Part I. The shock pair system, *Trans. R. Soc. London A,* (submitted).
22. L. F. Henderson, E. G. Puckett, and P. Colella, "On the Anomalous Refraction of Shock Waves," in *Proceedings, Second Japan-Soviet Union Symposium on Computational Fluid Dynamics, Tsukuba, Japan, 1990,* p. 144.
23. L. F. Henderson, E. G. Puckett, and P. Colella, Anomalous refraction of shock waves, in *Shock Waves,* edited by K. Takayama New York/Berlin, (Springer-Verlag, 1992), p. 283.
24. C. W. Hirt, *Flow-3D Users Manual* (Flow Sciences, Inc., 1988).
25. C. W. Hirt and B. D. Nichols, Volume of fluid (VOF) method for the dynamics of free boundaries, *J. Comput. Phys.* **39,** 201 (1981).
26. D. B. Kothe, J. R. Baumgardner, S. T. Bennion, J. H. Cerutti, B. J. Daly, K. S. Holian, E. M. Kober, S. J. Mosso, J. W. Painter, R. D. Smith, and M. D. Torrey," PAGOSA: A Massively-Parallel, Multi-Material Hydro-Dynamics Model for Three-Dimensional High-Speed Flow and High-Rate Deformation," Technical Report LA-UR-92-4306, Los Alamos National Laboratory, 1992 (unpublished).
27. D. B. Kothe and R. C. Mjolsness, RIPPLE: A new model for incompressible flows with free surfaces, *AIAA J.* **30**(11), 2694 (1992).
28. 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, April 1991.
29. B. LaFaurie, C. Nardone, R. Scardovelli, S. Zaleski, and G. Zanetti, Modelling merging and fragmentation in multiphase flows with SURFER, *J. Comput. Phys.* **113,** 134 (1994).
30. R. J. LeVeque, "High-Resolution Conservative Algorithms for Advection in Incompressible Flow," *SIAM J. Numer. Anal.* **33,** 627 (1996).
31. H. Liu, E. J. Lavernia, and R. H. Rangel, Numerical investigation of micropore formation during substrate impact of molten droplets in plasma spray processes, *Atomization Sprays* **4,** 369 (1994).
32. D. L. Marcus and J. B. Bell, Numerical simulation of a viscious vortex ring interaction with a density interface, *Phys. Fluids A* **6**(4), 1505 (1994).
33. D. L. Marcus, E. G. Puckett, J. B. Bell, and J. S. Saltzmann, "Numerical Simulation of Accelerated Interfaces," *3rd International Workshop on the Physics of Compressible Turbulent Mixing,* edited by R. Dautray (CEA DAM, 1991), p. 63.
34. W. Mulder, S. Osher, and J. A. Sethian, Computing interface motion in compressible gas dynamics, *J. Comput. Phys.* **100,** 209 (1992).
35. B. D. Nichols, C. W. Hirt, and R. S. Hotchkiss, "SOLA-VOF: A Solution Algorithm for Transient Fluid Flow with Multiple Free Boundaries," Technical Report LA-8355, Los Alamos National Laboratory, August 1980 (unpublished).
36. W. F. Noh and P. R. Woodward, "SLIC (Simple Line Interface Calculation)" in *Lecture Notes in Physics,* Vol. 59, edited by A. I. van der Vooren and P. J. Zandbergen (Springer-Verlag, New York/Berlin, 1976), p. 330.
37. H. N. Oguz and A. Prosperetti, Dynamics of bubble growth and detachment from a needle, *J. Fluid Mech.* **257,** 111 (1993).
38. S. Osher and J. A. Sethian, Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton–Jacobi formulations, *J. Comput. Phys.* **79**(1), 12 (1988).
39. B. J. Parker and D. L. Youngs, "Two and Three Dimensional Eulerian Simulation of Fluid Flow with Material Interfaces," Technical Report 01/92, UK Atomic Weapons Establishment, Aldermaston, Berkshire, Feb. 1992 (unpublished).
40. J. E. Pilliod, "An Analysis of Piecewise Linear Interface Reconstruction Algorithms for Volume-Of-Fluid Methods," Master's thesis, U.C. Davis, September 1992 (unpublished).
41. J. E. Pilliod and E. G. Puckett, Second-order volume-of-fluid interface tracking algorithms, *J. Comput. Phys.,* submitted.
42. E. G. Puckett, "A Volume-of-Fluid Interface Tracking Algorithm with Applications to Computing Shock Wave Refraction, in *Proceedings, 4th International Symposium on Computational Fluid Dynamics, Davis, CA, 1991,* edited by H. Dwyer, p. 933.
43. E. G. Puckett, L. F. Henderson, and P. Colella, The anomalous refraction of shock waves in materials with general equations of state. Part II. Anomalous refraction wave systems, *Trans. R. Soc. London A,* submitted.
44. E. G. Puckett, L. F. Henderson, and P. Colella, "A General Theory of Anomalous Refraction," in *Shock Waves @ Marseilles,* Vol. IV edited by R. Brun and L. Z. Dumitrescu (Springer-Verlag, New York/Berlin, 1995), p. 139.
45. E. G. Puckett and J. S. Saltzman, A 3-D adaptive mesh refinement algorithm for multimaterial gas dynamics, *Physica D* **60,** 84 (1992).
46. W. J. Rider, D. B. Kothe, S. J. Mosso, J. H. Cerutti, and J. I. Hochstein,

"Accurate Solution Algorithms for Incompressible Multiphase Flows," in *Proceedings, 33rd AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, 1995;* AIAA-95-0699.

47. J. A. Sethian, Turbulent combustion in open and closed vessels, *J. Comput. Phys.* **54,** 425 (1984).
48. J. A. Sethian, Numerical algorithms for propagating interfaces: Hamilton–Jacobi equations and conservation laws, *J. Differential Geom.* **31,** 131 (1990).
49. J. A. Sethian and J. Strain, Crystal growth and dendritic solidification, *J. Comput. Phys.* **98,** 231 (1992).
50. J. Strain, A boundary integral approach to unstable solidification, *J. Comput. Phys.* **85,** 342 (1989).
51. G. Strang, On the construction and comparison of difference schemes, *SIAM J. Numer. Anal.* **5,** 506 (1968).
52. M. Sussman, E. Fatemi, P. Smereka, and S. Osher, "A Level Set Approach for Computing Solutions to Incompressible Two-Phase Flow II," in *Proceedings, 6th International Symposium on Computational Fluid Dynamics, Lake Tahoe, Nevada, September 1995,* edited by M. Hafez.
53. M. Sussman, E. Fatemi, P. Smereka, and S. Osher, An improved level set method for incompressible two-phase flow, *J. Comput. & Fluids,* to appear.
54. M. Sussman, P. Smereka, and S. Osher, A level set approach for computing solutions to incompressible two-phase flow, *J. Comput. Phys.* **114**(1), 146 (1994).
55. O. Tatebe, "The Multigrid Preconditioned Conjugate Gradient Method," in *Proceedings, Sixth Copper Mountain Conference on Multigrid Methods, Copper Mountain, CO, April 4–9, 1993,* Vol. CP-3224, edited by N. D. Melson, T. A. Manteuffel, and S. F. McCormick, p. 621.
56. P. A. Torpey, "Prevention of Air Ingestion in a Thermal Ink-Jet Device," *Proceedings, 4th International Congress on Advances in Non-Impact Print Technologies, March 1988.*
57. M. D. Torrey, L. D. Cloutman, R. C. Mjolsness, and C. W. Hirt, "NASA-VOF2D: A Computer Program for Incompressible Flows with Free Surfaces," Technical Report LA-10612-MS, Los Alamos National Laboratory, December 1985 (unpublished).
58. M. D. Torrey, R. C. Mjolsness, and L. R. Stein, "NASA-VOF3D: A Three-Dimensional Computer Program for Incompressible Flows with Free Surfaces," Technical Report LA-11009-MS, Los Alamos National Laboratory, July 1987 (unpublished).
59. G. Trapaga, E. F. Matthys, J. J. Valencia, and J. Szekely, Fluid flow, heat transfer, and solidification of molten metal droplets impinging on substrates—Comparison of numerical and experimental results, *Metall. Trans. B.* **23**(6), 701 (1992).
60. G. Tryggvason, University of Michigan, Ann Arbor, MI, 1994, private communication.
61. S. O. Unverdi and G. Tryggvason, Computations of multi-fluid flows, *Physica D* **60,** 70 (1992).
62. J. K. Walters and J. F. Davidson, The initial motion of a gas bubble formed in an inviscid liquid. Part 1. The two-dimensional bubble, *J. Fluid Mech.* **12,** 408 (1962).
63. D. L. Youngs, Time-dependent multi-material flow with large fluid distortion, in *Numerical Methods for Fluid Dynamics,* edited by K. W. Morton and M. J. Baines (Academic Press, New York, 1982).
64. D. L. Youngs, "Numerical Simulation of Turbulent Mixing by Rayleigh-Taylor Instability," in *Fronts, Interfaces and Patterns,* edited by A. R. Bishop, L. J. Campbell, and P. J. Channell (North-Holland, Amsterdam, 1984), p. 32.