Journal of Computational Physics **157,** 441–472 (2000)
doi:10.1006/jcph.1999.6348, available online at http://www.idealibrary.com on IDEAL®

Numerical Simulation of Axisymmetric Free Surface Flows

M. F. Tome,* A. Castelo,† J. Murakami,† J. A. Cuminato,† R. Minghim, M. C. F. Oliveira,† N. Mangiavacchi,† and S. McKee‡

**Department of Mathematics, Instituto Superior Técnico, Lisbon, Portugal; †ICMC—Campus de São Carlos, University of São Paulo, Brazil; and ‡Department of Mathematics, University of Strathclyde, Glasgow, Scotland*

Received October 15, 1997; revised May 6, 1999

This paper describes an extension of the GENSMAC code for solving two-dimensional free surface flows to axisymmetric flows. Like GENSMAC the technique is finite difference based and embodies, but considerably extends, the SMAC (simplified marker and cell) ideas. It incorporates adaptive time stepping and an accurate representation of the free surfaces while at the same time only uses surface particles to define the free surfaces, greatly increasing the computational speed; in addition, it employs a graphic interface with solid modeling techniques to provide enhanced three-dimensional visualization. Various simulations are undertaken to illustrate and validate typical flows. Both G. I. Taylor's viscous jet plunging into a fluid and a liquid drop splashing onto a fluid are simulated. Also, the important industrial application of container filling is illustrated. Finally, a comparison is made with the linear theory of standing waves and the code is validated by a numerical convergence study. © 2000 Academic Press

Key Words: Navier–Stokes; marker-and-cell method; free-surface stress conditions; finite difference method; axisymmetric flows.

1. INTRODUCTION

Industrial applications of fluid flows with free surfaces are ubiquitous: applications include casting, container filling, extrusion, and fluid jetting devices. The accurate determination of these free surfaces is important especially if the flow itself is determined by the position and curvature of the free surface as it would be if surface tension were significant. It is also essential that any numerical algorithm can cope with merging, folding, or separation of free surfaces.

Over the years a number of computational techniques have been developed for solving free surface flows (see, e.g., Shyy *et al.* [1]). These may be broadly divided into two categories: interface tracking methods and front-capturing methods. Front or shock-capturing methods are usually associated with compressible fluids; these methods are now extremely sophisticated, explicitly enforcing monotonicity through a nonlinear step while simultaneously maintaining high order. The reader is referred, for instance, to the books by LeVeque [2] and Hirsch [3].

Tracking methods may be subdivided into front-tracking and volume-tracking. If high accuracy is required it is generally accepted that front-tracking is needed, when the interface itself is described by additional computational elements. Although the basic idea goes back to Richtmyer and Morton [4], its primary implementation has been through the work of Glimm and his co-workers (see, e.g., Glimm *et al.* [5]). They represent the moving front by a connected set of points which form a moving internal boundary. To calculate the evolution inside the fluid in the vicinity of the interface, an irregular grid is constructed and a special finite difference stencil is used on these irregular grids. Authors who have used this approach to different flow regimes include Chern *et al.* [6], Daripa *et al.* [7], Moretti [8], and Peskin [9] (also Fauci and Peskin [10] and Fogelson and Peskin [11]). In Peskin's work the connected set of particles carry forces which are adjusted to achieve a specific velocity at the interface. Within this category one might include the so-called boundary integral or boundary element methods and the vortex-in-cell (VIC) method. Boundary integral methods can be effective when inertia forces are negligible (see, for instance, Baker and Moore [12] or Tsai and Miksis [13] who solve successfully the axisymmetric problem of gas bubbles rising in a liquid). The vortex-in-cell method, normally used for homogeneous flows, has been extended to cope with weakly stratified flows (Meng and Thomson [14]) and arbitrary stratification (Tryggvason [15]). More recently Unverdi and Tryggvason [16] described a front-tracking method for incompressible, viscous, multifluid flows in which the interface is explicitly tracked but maintains a distinct thickness dependent upon the mesh size. The main advantage of this approach is that interfaces can interact in a rather natural way, since gradients simply add or cancel as the grid distribution is constructed from the information carried by the tracked front. Another approach which has found favor is the level set approach. This would appear to have been first introduced by Osher and Sethian [17]. The level set function is typically a smooth function which eliminates the sorts of problems, such as oscillations, that conventional difference schemes often have. It also removes having to add or subtract points to a moving grid and it automatically takes care of merging and breaking up of an interface. More recently, Sussman *et al.* [18] have combined the level set approach with projection methods (see, e.g., Bell and Marcus [19]) to avoid explicitly tracking the interface. A level set approach has also been applied to three-dimensional two-phase flows by Beaux and Banerjee [20].

Volume-tracking methods can be further subdivided into marker-and-cell (MAC) and volume-of-fluid (VOF) methods. Indeed, the original MAC method was one of the first such tracking methods dating back to Harlow and Welch [21]. Both these classes of methods are still popular and, although they suffer from not being able to accurately provide a surface interface, arguably this is less important today—a $100 \times 100 \times 100$ grid is possible on a good workstation and will certainly be easily feasible on even a modest one in the next few years. With the MAC method virtual marker particles are pushed forward according to the Eulerian fluid calculation (with appropriate bilinear interpolation for the velocity components) and it is these that define the fluid region and hence the interface. The simplified-marker-and-cell (SMAC) was introduced by Amsden and Harlow [22]. Over the intervening years research into this method has continued; see, for example, Miyata [23], Viecelli [24], and Hirt and

Shannon [25] who also used the immersed boundary technique to handle its interaction with the underlying grid.

Possibly the first volume-in-fluid type code was the simple line interface calculation (SLIC) of Noh and Woodward [27]. This was employed by Chorin [28] to model flame propagation and later by Ghoniem *et al.* [29] and Sethian [30] to model turbulent combustion. However, one usually associates Hirt and Nichols [31] with the VOF method, whereby a volume fraction is convected forward with the fluid. This then led to many variants and descendants, namely, SOLA-VOF (Nichols *et al.* [32]), NASA-VOF 2D (Torrey *et al.* [33]), NASA-VOF3D (Torrey *et al.* [34]), RIPPLE (Kothe and Mjolsness [35], Koth *et al.* [36]) and Flow3D (Hirt [37]). These have been widely used in industrial applications. Most recently, an interesting idea of a second order VOF tracking method, employing an approximate projection operator, has been put forward by Puckett *et al.* [38].

Recently, Tomé and McKee [39], motivated by industrial filling processes, returned to the SMAC methodology and developed the GENSMAC code. GENSMAC simulates incompressible time dependent fluid flows in Cartesian coordinates within arbitrary, user-specified two-dimensional domains. In addition, it can handle free-slip and no-slip boundary conditions, there can be a number of inflows and outflows, and a number of arbitrary shaped obstacles can be contained within the general flow domain. However, although GENSMAC has a wide range of applicability it cannot deal with axisymmetric flows nor can it display output in a three-dimensional form. This paper describes briefly how GENSMAC may be modified to cope with axisymmetric flows; it also discusses how the techniques of solid modeling may be applied through a graphic interface to permit enhanced flow visualization.

2. BASIC EQUATIONS

We consider incompressible axisymmetric Newtonian flows. The governing equations are the nondimensional mass and momentum equations in conservative form which in cylindrical coordinates may be written as [22]

$$\frac{1}{r}\frac{\partial(ru)}{\partial r}+\frac{\partial v}{\partial z}=0 \tag{1}$$

$$\frac{\partial u}{\partial t}+\frac{1}{r}\frac{\partial(ru^2)}{\partial r}+\frac{\partial(uv)}{\partial z}=-\frac{\partial p}{\partial r}+\frac{1}{\mathrm{Re}}\frac{\partial}{\partial z}\left(\frac{\partial u}{\partial z}-\frac{\partial v}{\partial r}\right)+\frac{1}{\mathrm{Fr}^2}g_r \tag{2}$$

$$\frac{\partial v}{\partial t}+\frac{1}{r}\frac{\partial(ruv)}{\partial r}+\frac{\partial v^2}{\partial z}=-\frac{\partial p}{\partial z}-\frac{1}{\mathrm{Re}}\frac{1}{r}\frac{\partial}{\partial r}\left(r\left(\frac{\partial u}{\partial z}-\frac{\partial v}{\partial r}\right)\right)+\frac{1}{\mathrm{Fr}^2}g_z, \tag{3}$$

where $\mathrm{Re}=UL/\nu$ and $\mathrm{Fr}=U/\sqrt{Lg}$ denote the Reynolds number, and the Froude number, respectively. Here L and U are the length and velocity scales, respectively, ν is a reference viscosity, and g denotes the gravitational constant, $g=|\mathbf{g}|$, where $\mathbf{g}=(g_r, g_z)$. Furthermore, $\mathbf{u}=(u, v)^T$ are the radial and vertical components of velocity while p is the pressure per unit density.

3. METHOD OF SOLUTION

In order to solve Eqs. (1)–(3) we employ the GENSMAC methodology. In particular, in calculating $\tilde{\mathbf{u}}(r, z, t)$ in step 2 we employ an efficient adaptive time stepping routine. A more complete description may be found in [39].

It is supposed that at a given time t_0, the velocity field $\mathbf{u}(r, z, t_0)$ is known and boundary conditions for the velocity and pressure are given. The updated velocity field $\mathbf{u}(r, z, t)$ at $t = t_0 + \delta t$ is calculated as follows:

1. Let $\tilde{p}(r, z, t)$ be a pressure field which satisfies the correct pressure condition on the free surface. This pressure field is computed according to the stress conditions given in Section 4.2.

2. Calculate the intermediate velocity field $\tilde{\mathbf{u}}(r, z, t)$ from the explicitly discretized form of

$$\frac{\partial \tilde{u}}{\partial t} = \left[-\frac{1}{r}\frac{\partial (ru^2)}{\partial r} - \frac{\partial (uv)}{\partial z} - \frac{\partial \tilde{p}}{\partial r} + \frac{1}{\mathrm{Re}}\frac{\partial}{\partial z}\left(\frac{\partial u}{\partial z} - \frac{\partial v}{\partial r}\right) + \frac{1}{\mathrm{Fr}^2} g_r \right]_{t=t_0} \tag{4}$$

$$\frac{\partial \tilde{v}}{\partial t} = \left[-\frac{1}{r}\frac{\partial (ruv)}{\partial r} - \frac{\partial (v^2)}{\partial z} - \frac{\partial \tilde{p}}{\partial z} - \frac{1}{\mathrm{Re}}\frac{1}{r}\frac{\partial}{\partial r}\left(r\left(\frac{\partial u}{\partial z} - \frac{\partial v}{\partial r}\right)\right) + \frac{1}{\mathrm{Fr}^2} g_z \right]_{t=t_0} \tag{5}$$

with $\tilde{\mathbf{u}}(r, z, t_0) = \mathbf{u}(r, z, t_0)$ using the correct boundary conditions for $\mathbf{u}(r, z, t_0)$. It can be shown [40] that $\tilde{\mathbf{u}}(r, z, t)$ possesses the correct vorticity at time t. However, $\tilde{\mathbf{u}}(r, z, t)$ does not satisfy (1). Let

$$\mathbf{u}(r, z, t) = \tilde{\mathbf{u}}(r, z, t) - \nabla\psi(r, z, t) \tag{6}$$

with

$$\nabla^2\psi(r, z, t) = \nabla \cdot \tilde{\mathbf{u}}(r, z, t). \tag{7}$$

Thus, $\mathbf{u}(r, z, t)$ now satisfies (1) and the vorticity remains unchanged. Therefore, $\mathbf{u}(r, z, t)$ is identified as the updated velocity field at time t.

3. Solve the Poisson equation (7).
4. Compute the velocity (6).
5. Compute the pressure. It can be shown [40] that the pressure is given by

$$p(r, z, t) = \tilde{p}(r, z, t) + \psi(r, z, t)/\delta t. \tag{8}$$

6. Update the positions of the marker particles.

The last step in the calculation involves moving the marker particles to their new positions. These are virtual particles whose coordinates are stored and updated at the end of each cycle by solving

$$\frac{dr}{dt} = u, \quad \frac{dz}{dt} = v$$

by Euler's method. This provides a particle with its new coordinates, allowing us to determine whether or not it moved to a new computational cell or if it left the containment region through an outlet. Only marker particles on the surface are considered; Section 4.3 will describe how this is achieved.

3.1. *Boundary Conditions*

Boundary conditions must be imposed both on fixed boundaries and on free surfaces. On fixed boundaries we can impose no-slip, free-slip, prescribed inflow, prescribed outflow, and continuative outflow (for details, see [39, 41]). The implementation of these boundary conditions is performed in the same way as in the GENSMAC code.

The appropriate free surface boundary conditions are the vanishing of the normal and the tangential stresses which in the absence of surface tension are (see Batchelor [42, p. 150])

$$\mathbf{n} \cdot \sigma \cdot \mathbf{n} = 0 \tag{9}$$

$$\mathbf{m} \cdot \sigma \cdot \mathbf{n} = 0, \tag{10}$$

where **n** and **m** are local unit normal and tangential vectors and σ is the stress tensor given by

$$\sigma_{ij} = -p\delta_{ij} + \frac{2}{\mathrm{Re}}\left(\frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i}\right).$$

These conditions are applied by making accurate local finite difference approximations on the free surface [39] and will be given in Section 4.2. The appropriate boundary conditions for the Poisson equation (7) (see [22]) are

$$\frac{\partial \psi}{\partial n} = 0 \quad \text{on fixed boundaries and} \quad \psi = 0 \quad \text{on the free surface.} \tag{11}$$

4. BASIC FINITE DIFFERENCE EQUATIONS

To implement the method presented in Section 3 we employ the finite difference method as follows.

A staggered grid is employed. A typical cell is as shown in Fig. 1. The variables, pressure $\tilde{p}_{i,j}$ and the added velocity potential $\psi_{i,j}$, are positioned at the cell center while $u_{i,j}$ and $v_{i,j}$ are staggered by a translation of $\delta r/2$ and $\delta z/2$, respectively.

The momentum equations (4) and (5) are discretized and applied at the u-nodes and v-nodes, respectively. A forward difference in time is used for the time derivatives and the linear spatial terms on the right-hand side are approximated by central differences; for the convection terms in (4) and (5) the ZIP (see [22]) form is adopted. For the flux terms (uv),

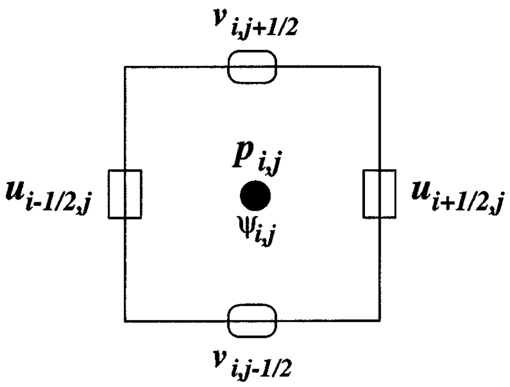


FIG. 1. Computational cell.

simple averages are performed; for instance,

$$(uv)_{i+\frac{1}{2},j+\frac{1}{2}} = \left(\frac{u_{i+\frac{1}{2},j} + u_{i+\frac{1}{2},j+1}}{2}\right)\left(\frac{v_{i,j+\frac{1}{2}} + v_{i+1,j+\frac{1}{2}}}{2}\right).$$

Thus, the finite difference approximations to (4) and (5) become [22]

$$\begin{aligned}\tilde{u}_{i+\frac{1}{2},j} = u_{i+\frac{1}{2},j} + \delta t\Bigg[&\frac{u_{i+\frac{1}{2},j}}{r_{i+\frac{1}{2}}\delta r}\left(r_i u_{i-\frac{1}{2},j} - r_{i+1}u_{i+\frac{3}{2},j}\right) + \frac{\tilde{p}_{i,j} - \tilde{p}_{i+1,j}}{\delta r}\\ &+ \frac{1}{\mathrm{Fr}^2}g_r + \frac{1}{4\delta z}\left(\left(u_{i+\frac{1}{2},j} + u_{i+\frac{1}{2},j-1}\right)\left(v_{i,j-\frac{1}{2}} + v_{i+1,j-\frac{1}{2}}\right)\right.\\ &\left.- \left(u_{i+\frac{1}{2},j} + u_{i+\frac{1}{2},j+1}\right)\left(v_{i,j+\frac{1}{2}} + v_{i+1,j+\frac{1}{2}}\right)\right)\\ &+ \frac{1}{\mathrm{Re}}\left(\frac{u_{i+\frac{1}{2},j+1} + u_{i+\frac{1}{2},j-1} - 2u_{i+\frac{1}{2},j}}{\delta z^2} - \frac{v_{i+1,j+\frac{1}{2}} - v_{i+1,j-\frac{1}{2}} - v_{i,j+\frac{1}{2}} + v_{i,j-\frac{1}{2}}}{\delta r\,\delta z}\right)\Bigg],\end{aligned} \tag{12}$$

$$\begin{aligned}\tilde{v}_{i,j+\frac{1}{2}} = v_{i,j+\frac{1}{2}} + \delta t\Bigg[&\frac{v_{i,j+\frac{1}{2}}\left(v_{i,j-\frac{1}{2}} - v_{i,j+\frac{3}{2}}\right)}{\delta z} + \frac{\tilde{p}_{i,j} - \tilde{p}_{i,j+1}}{\delta z} + \frac{1}{\mathrm{Fr}^2}g_z\\ &+ \frac{1}{4r_i\delta r}\left(r_{i-\frac{1}{2}}\left(u_{i-\frac{1}{2},j} + u_{i-\frac{1}{2},j+1}\right)\left(v_{i-1,j+\frac{1}{2}} + v_{i,j+\frac{1}{2}}\right)\right.\\ &\left.- r_{i+\frac{1}{2}}\left(u_{i+\frac{1}{2},j} + u_{i+\frac{1}{2},j+1}\right)\left(v_{i+1,j+\frac{1}{2}} + v_{i,j+\frac{1}{2}}\right)\right)\\ &- \frac{1}{\mathrm{Re}}\frac{1}{r_i\delta r}\left(r_{i+\frac{1}{2}}\left(\frac{u_{i+\frac{1}{2},j+1} - u_{i+\frac{1}{2},j}}{\delta z} - \frac{v_{i+1,j+\frac{1}{2}} - v_{i,j+\frac{1}{2}}}{\delta r}\right)\right.\\ &\left.- r_{i-\frac{1}{2}}\left(\frac{u_{i-\frac{1}{2},j+1} - u_{i-\frac{1}{2},j}}{\delta z} - \frac{v_{i,j+\frac{1}{2}} - v_{i-1,j+\frac{1}{2}}}{\delta r}\right)\right)\Bigg].\end{aligned} \tag{13}$$

Here r_i denotes $i\delta r$. The Poisson equation (7) in cylindrical coordinates becomes

$$\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial\psi}{\partial r}\right) + \frac{\partial^2\psi}{\partial z^2} = \tilde{D},$$

where

$$\tilde{D} = \frac{1}{r}\frac{\partial(r\tilde{u})}{\partial r} + \frac{\partial\tilde{v}}{\partial z}. \tag{14}$$

We point out that direct discretization of (14) would lead to a nonsymmmetric linear system which would be required to be solved by, for example, the biconjugate gradient method which is considerably less efficient than the conjugate gradient method. However, by rewriting (14) in the form

$$\frac{\partial}{\partial r}\left(r\frac{\partial\psi}{\partial r}\right) + r\frac{\partial^2\psi}{\partial z^2} = r\tilde{D} \tag{15}$$

and discretizing (15) at the surface cell center we obtain (assuming $\delta r = \delta z$)

$$-r_i\psi_{i,j-1} - r_{i-\frac{1}{2}}\psi_{i-1,j} + 4r_i\psi_{i,j} - r_{i+\frac{1}{2}}\psi_{i+1,j} - r_i\psi_{i,j+1} = -r_i\delta r^2\tilde{D}_{i,j}, \tag{16}$$

where

$$\tilde{D}_{i,j} = \frac{1}{r_i}\left(\frac{r_{i+\frac{1}{2}}\tilde{u}_{i+\frac{1}{2},j} - r_{i-\frac{1}{2}}\tilde{u}_{i-\frac{1}{2},j}}{\delta r}\right) + \frac{\tilde{v}_{i,j+\frac{1}{2}} - \tilde{v}_{i,j-\frac{1}{2}}}{\delta z}.$$

It is now easily seen that (16) leads to a linear system possessing a symmmetric positive definite matrix and consequently we employ the conjugate gradient method as implemented in GENSMAC for solving this linear system.

4.1. *Cell Flagging*

As the fluid is continuously moving, a procedure for identifying the fluid region and the free surface is employed. To accommodate this, the cells within the mesh are flagged according to whether they are surface cells (S), full cells (F), empty cells (E), boundary cells (B), inflow cells (I), or outflow cells (O). A detailed description can be found in [39].

4.2. *Free Surface Stress Conditions*

The stress conditions (9), (10) can be written as

$$p - \frac{2}{\text{Re}}\left[\frac{\partial u}{\partial r}n_r^2 + \frac{\partial v}{\partial z}n_z^2 + \left(\frac{\partial u}{\partial z} + \frac{\partial v}{\partial r}\right)n_r n_z\right] = 0, \tag{17}$$

$$\frac{2}{\text{Re}}\left[\left(\frac{\partial u}{\partial r} - \frac{\partial v}{\partial z}\right)n_r n_z + \left(\frac{\partial u}{\partial z} + \frac{\partial v}{\partial r}\right)\left(n_r^2 - n_z^2\right)\right] = 0. \tag{18}$$

In order to apply these conditions we follow the approximations adopted by GENSMAC [39]; namely, we consider two types of free surface orientations as follows:

(a) Horizontal/vertical surfaces: These surfaces are identified by surface cells having only one side contiguous with empty cells. For these cells we assume that the normal vector is pointing toward the empty cell in which case we take $\mathbf{n} = (n_r, 0)$ or $\mathbf{n} = (0, n_z)$. The choice is made according to which side is contiguous with the empty cell. For instance, if a surface cell has only the top side contiguous with an empty cell (see Fig. 2), then we take

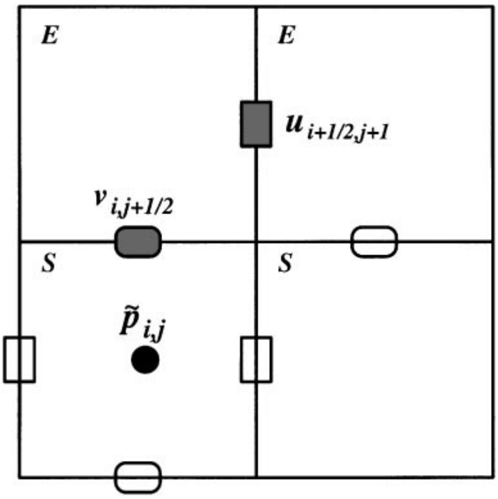


FIG. 2. Surface cell with only the top side contiguous with an empty cell.

$\mathbf{n} = (0, 1)$. Equations (17) and (18) then reduce to

$$p - \frac{2}{\mathrm{Re}}\left(\frac{\partial v}{\partial z}\right) = 0, \tag{19}$$

$$\left(\frac{\partial u}{\partial z} + \frac{\partial v}{\partial r}\right) = 0. \tag{20}$$

It can be seen that when computing the tilde velocities through (12) and (13) the pressure $\tilde{p}_{i,j}$ and the values of $u_{i+\frac{1}{2},j+1}$ and $v_{i,j+\frac{1}{2}}$ at the empty cell faces are required (see Fig. 2). These can be obtained from (19), (20) and the mass conservation equation (1) as follows. First, by applying (1) at the surface cell center we get

$$\frac{1}{r_i}\left(\frac{r_{i+\frac{1}{2}}u_{i+\frac{1}{2},j} - r_{i-\frac{1}{2}}u_{i-\frac{1}{2},j}}{\delta r}\right) + \frac{v_{i,j+\frac{1}{2}} - v_{i,j-\frac{1}{2}}}{\delta z} = 0$$

which gives

$$v_{i,j+\frac{1}{2}} = v_{i,j-\frac{1}{2}} - \frac{\delta z}{\delta r}\frac{1}{r_i}\left(r_{i+\frac{1}{2}}u_{i+\frac{1}{2},j} - r_{i-\frac{1}{2}}u_{i-\frac{1}{2},j}\right).$$

Now, discretizing (20) at the position $(i+\frac{1}{2}, j+\frac{1}{2})$ we obtain

$$\frac{u_{i+\frac{1}{2},j+1} - u_{i+\frac{1}{2},j}}{\delta z} + \frac{v_{i+1,j+\frac{1}{2}} - v_{i,j+\frac{1}{2}}}{\delta r} = 0$$

which yields

$$u_{i+\frac{1}{2},j+1} = u_{i+\frac{1}{2},j} - \frac{\delta z}{\delta r}\left(v_{i+1,j+\frac{1}{2}} - v_{i,j+\frac{1}{2}}\right).$$

Once the velocities have been computed the pressure $\tilde{p}_{i,j}$ follows from (19) applied at the surface cell center, giving

$$\tilde{p}_{i,j} = \frac{2}{\mathrm{Re}}\left(\frac{v_{i,j+\frac{1}{2}} - v_{i,j-\frac{1}{2}}}{\delta z}\right).$$

Other types of configurations of surface cells having only one side contiguous with empty cells are treated similarly.

(b) 45°-sloped surface: These surfaces are identified by surface cells having two adjacent faces contiguous with empty cells. For these cells we assume that the normal vector makes 45° with the axes in which case we take $\mathbf{n} = (\pm\frac{\sqrt{2}}{2}, \pm\frac{\sqrt{2}}{2})$. The sign is chosen according to which faces are contiguous with empty cells. For instance, let us consider the surface cell in Fig. 3. For this cell we take $\mathbf{n} = (\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2})$ and introducing it into (17) and (18), we obtain

$$p - \frac{1}{\mathrm{Re}}\left[\frac{\partial u}{\partial r} + \frac{\partial v}{\partial z} + \left(\frac{\partial u}{\partial z} + \frac{\partial v}{\partial r}\right)\right] = 0, \tag{21}$$

$$\left(\frac{\partial u}{\partial r} - \frac{\partial v}{\partial z}\right) = 0. \tag{22}$$

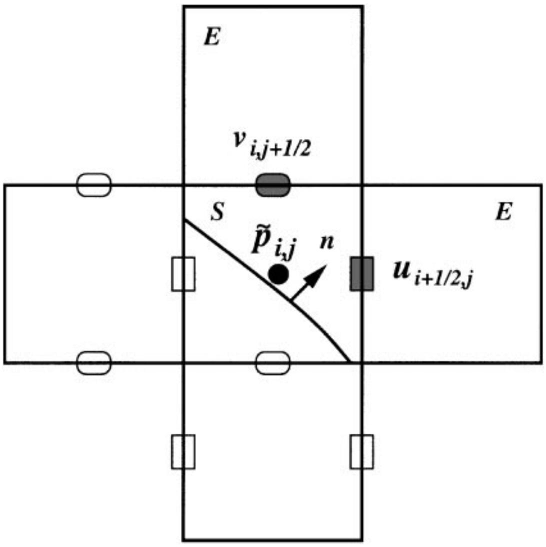


FIG. 3. Surface cell having the top and right sides contiguous with empty cells.

As we can see in Fig. 3, the values of $u_{i+\frac{1}{2},j}$ and $v_{i,j+\frac{1}{2}}$ are required. These are obtained by applying (22) and the mass conservation equation (1) at the surface cell center to give

$$\frac{u_{i+\frac{1}{2},j}-u_{i-\frac{1}{2},j}}{\delta r}+\frac{v_{i,j+\frac{1}{2}}-v_{i,j-\frac{1}{2}}}{\delta z}=0, \tag{23}$$

$$\frac{1}{r_i}\left(\frac{r_{i+\frac{1}{2}}u_{i+\frac{1}{2},j}-r_{i-\frac{1}{2}}u_{i-\frac{1}{2},j}}{\delta r}\right)+\frac{v_{i,j+\frac{1}{2}}-v_{i,j-\frac{1}{2}}}{\delta z}=0. \tag{24}$$

Equations (23) and (24) provide a (2×2)-linear system for $u_{i+\frac{1}{2},j}$ and $v_{i,j+\frac{1}{2}}$, yielding

$$u_{i+\frac{1}{2},j}=\frac{r_i+r_{i-\frac{1}{2}}}{r_i+r_{i+\frac{1}{2}}}u_{i-\frac{1}{2},j}, \tag{25}$$

$$v_{i,j+\frac{1}{2}}=v_{i,j-\frac{1}{2}}+\frac{\delta z}{\delta r}\left(u_{i+\frac{1}{2},j}-u_{i-\frac{1}{2},j}\right). \tag{26}$$

Once the velocities at the empty cell faces have been computed, the pressure follows from (21) applied at the surface cell center, giving

$$\begin{aligned}\tilde{p}_{i,j}=\frac{1}{\mathrm{Re}}\Bigg[&\frac{u_{i+\frac{1}{2},j}-u_{i-\frac{1}{2},j}}{\delta r}+\frac{v_{i,j+\frac{1}{2}}-v_{i,j-\frac{1}{2}}}{\delta z}\\&+\frac{1}{2}\left(\frac{u_{i+\frac{1}{2},j}+u_{i-\frac{1}{2},j}-u_{i+\frac{1}{2},j-1}-u_{i-\frac{1}{2},j-1}}{\delta z}\right.\\&\left.+\frac{v_{i,j+\frac{1}{2}}+v_{i,j-\frac{1}{2}}-v_{i-1,j+\frac{1}{2}}-v_{i-1,j-\frac{1}{2}}}{\delta r}\right)\Bigg].\end{aligned} \tag{27}$$

For other configurations of surface cells with two adjacent sides contiguous with empty cells the values of u, v, and $\tilde{p}$ are obtained similarly.

(c) Surface cells with three sides or two opposite sides contiguous with empty cells: These cells do not provide enough information to obtain an approximation for the unit normal **n**. If they appear during a calculation we set the pressure equal to zero and adjust at least one velocity on one of the empty cell faces so that mass is conserved. To minimize the appearance of such cells a finer mesh should be employed.

4.3. *Particle Movement*

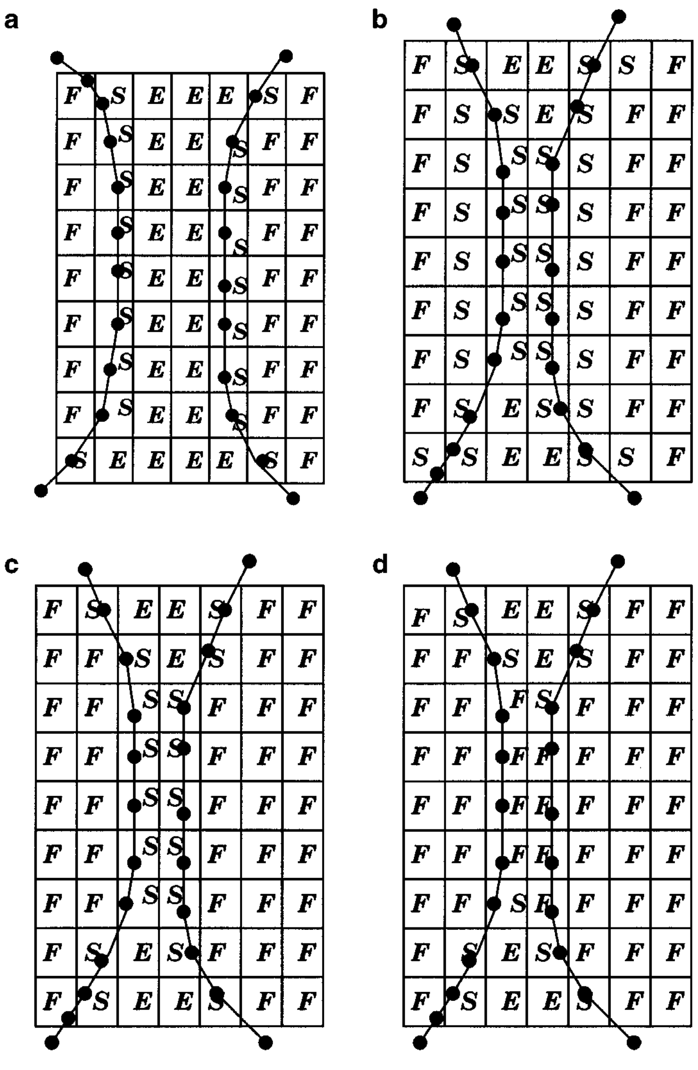
In order to improve the computational efficiency of the code, individual modules or subprograms making up GENSMAC were analyzed. It was found that the particle movement routine accounted for 40% of the total computational time. This is principally because a large number of particles, often as many as 20 per cell, are required to accurately represent the fluid. In truth, internal particles are redundant since the surface particles define the boundary of the fluid and thus the fluid region itself. Thus, a new technique was devised based upon representing the fluid by its boundary using a set of ordered lists defining the interior (and hence the exterior) of the fluid region. Each list stores connected information about the position of the particle, the type of cell it is located in, and the type of movement the particle is entitled to make. For instance, a node of type "inflow" cannot move, whereas a node of type "surface" can move freely according to the velocity field. The fluid movement is obtained by solving $\dot{r} = u(r, z, t)$, $\dot{z} = v(r, z, t)$, where u, v are the velocities in the r- and z-directions, respectively. As u and v are defined on a staggered grid, the velocity in each node is obtained from bilinear interpolation using the four nearest velocities. From time to time neighboring particles get too close or become separated by too great a distance. In the first case it is necessary to have a systematic means of particle removal while in the second if a surface cell becomes devoid of particles, a new particle is created in that cell equidistant from its two nearest neighbors. This new technique, simple though it is, has enhanced the computational efficiency of the code enabling it to solve both the jet flow problem and the splashing drop problem in reasonable time on a workstation.

4.4. *Merging of Surfaces*

As mentioned in Section 4.1, at each calculational cycle a reflagging algorithm is performed which updates the cells in the mesh. This reflagging algorithm is based on the following three steps:

1. During the particle movement all cells into which particles have moved are flagged as **S** cells. Observe that during this step the passage $\mathbf{F} \rightarrow \mathbf{S}$ and $\mathbf{E} \rightarrow \mathbf{S}$ is performed (see Scheme 1b).
2. A sweep is made on the surface cells which do not contain particles (see Scheme 1c). If a surface cell has no face which is contiguous with an empty cell then it becomes a full cell; otherwise, it becomes an empty cell. In this step the passage $\mathbf{S} \rightarrow \mathbf{F}$ and $\mathbf{S} \rightarrow \mathbf{E}$ is performed.
3. A sweep is made on the surface cells which contain particles (see Scheme 1d). If a surface cell does not have any face contiguous with a empty cell then it becomes a full cell. In this step the passage $\mathbf{S} \rightarrow \mathbf{F}$ is performed.

Scheme 1 displays one step of particle evolution and illustrates how free surfaces are allowed to merge.



SCHEME 1. Configuration of the cells in the mesh: (a) before particle movement, (b) after particle movement, (c) after application of step 2 of reflagging, (d) after application of step 3 of reflagging.

5. IMPLEMENTATION NOTES

The inclusion of cylindrical coordinates into the GENSMAC code will only affect steps 1, 2, and 3 in the computational procedure given in Section 3. Steps 4, 5, and 6 remain the same since the equations do not change from those of two-dimensional Cartesian flows. It was therefore necessary to write a routine to calculate the tilde pressure field $\tilde{p}(r, z, t)$ from the stress conditions on the free surface (see Section 4.2); it was further necessary to write a routine to calculate the tilde velocities, $\tilde{u}(r, z, t)$, $\tilde{v}(r, z, t)$ from Eqs. (12) and (13). On the other hand the discretized Poisson equation can be solved by the same conjugate gradient technique employed in GENSMAC. Thus, only a routine to assemble the matrix and the right-hand-side vector was required. Finally, it is necessary to obtain boundary conditions for $\mathbf{u}(r, z, t)$ on the free surface: this was achieved by a routine which computed the velocities on empty cell faces using the equations given in Section 4.2. The boundary conditions on rigid boundaries were handled in exactly the same way as for the original GENSMAC code [39].

6. VISUALIZATION AND SOLID MODELING

The amount of data generated by modern CFD codes is such as to make flow visualization essential. The visualization package associated with GENSMAC has the facility to collect domain data and flow parameters and display them graphically.

In this paper a specific package (VisFreeFlow) was developed for modeling the flow domain, for parameter reading, and for visualization of the results. The module for the flow domain is achieved by means of a graphical editor comprising a drawing area, command buttons for interacting with the model, configuration options, and file management tools. Each fluid containment region may be designed using polylines and arcs. Inflows and outflows are specified and displayed using different colors. Additional interaction techniques such as “dragging,” “rubberbanding,” and grid referencing have been made available. Domain data can also be input as a connected coordinate list in the input file accepted by GENSMAC. The domain editor works independent of the simulation process, thereby allowing, for example, the same mould to be used in connection with distinct simulations. The parameter reading module of VisFreeFlow provides a graphical user interface for inputing data such as viscosity, length, and velocity scales. This module is based on the XWindows windowing system, with motif-like dialogue structures, and has the following features readily available: data validation, data filing, data retrieving, and the changing of flow data options. It is also responsible for triggering the simulation process.

The output of the simulation can be graphically displayed with the visualization module of VisFreeFlow. The visualizations that are possible include the cell grid, the velocity, and the pressure. The fluid flow itself is visualized by boundary tracing and space filling. Pressure can be visualized either using isolines or by color mapping of the pressure ranges. The velocity field can be visualized either by vector plots that represent direction and magnitude or by using isolines in a manner similar to that used for the pressure plots. The parameters for all of the visualization options may be interactively changed to obtain the best resolution. Multiple viewing is also available by using different windows or by “layering” plots together. When detailed analysis is required a zoom in and out facility is available. In addition, VisFreeFlow has animation facilities.

The VisFreeFlow software, described above, is restricted to demonstrating two-dimensional fluid flow. However, the simulation data employed by VisFreeFlow can be used by the visualization package VTK—The Visualization Toolkit [44]—to generate three-dimensional visualizations (see Figs. 7–11, 14, and 15). VTK provides a portable set of data formats and visualization algorithms to operate on the data. The package can be tailored for user specific applications by building an appropriate interface. It is object-orientated and can be programmed using C++ or Tcl/Tk. By taking advantage of the axisymmetric nature of the free surface flows under consideration, three-dimensional visualizations were generated by rotating about the z-axis the available data defining the two-dimensional surface. The data from the two-dimensional fluid flow simulation were converted from their original format into a suitable VTK data format through a C program [45]. The VTK primitives used to describe the container, the extrudor (where appropriate), and the fluid surface at different times are polylines. For example, the polylines describing the two-dimensional profiles of the container, the extrudor, and the fluid surface at each instant of time were generated as

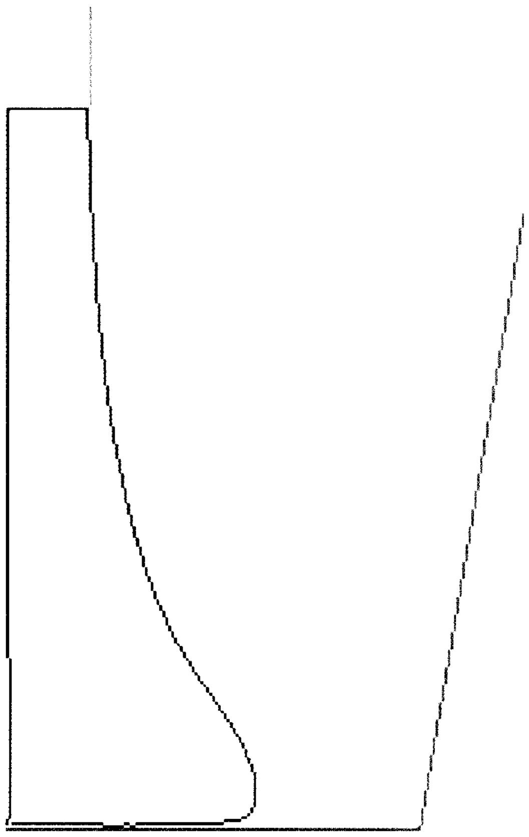


FIG. 4. Primitives used to generate the three-dimensional view of the cavity filling problem (see Fig. 11).

illustrated in Fig. 4. The profiles for the jet flow, the splashing drop and the cavity-filling simulations at different times (see Section 7) were rotated about the z-axis, generating the three-dimensional visualization shown in Figs. 6–10, 13, and 14, respectively. Each picture displays the fluid flow configuration at a different time. The complete set of images may be recorded as a video file for animation of the flow simulation. Details on the data conversion, file formats, and visualization can be found in [45].

7. NUMERICAL EXAMPLES AND QUALITATIVE VALIDATION

In this section we shall consider three problems: a vertical jet plunging into a fluid, a splashing drop, and container filling by extrusion from a circular nozzle.

7.1. *Jet Flow Experiment*

In an important early work Taylor [43] experimented with jet flows. In particular, he injected a jet into a box containing the same fluid and this he did for four different fluids. He identified the jet by coloring it while leaving the fluid in the box uncolored. By maintaining the nozzle diameter D and the inlet velocity V constant (see Fig. 5), any differences in the flows were only due to differences in the Reynolds number, the Froude number remaining

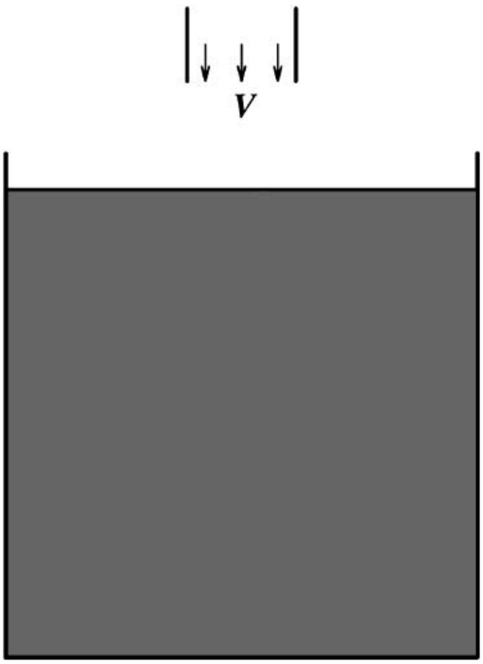


FIG. 5. Jet flow experiment.

constant throughout. The experiments were performed for the following Reynolds numbers: Re = 0.05, 10, 200, and 3000. It was seen that in the case of Re = 0.05 the simulated highly viscous fluid produced a compression stress causing the jet to buckle on reaching the free surface of the quiescent fluid and so not penetrate the mass of fluid contained within the box. For Re = 3000 the jet quickly broke up, causing rapid mixing and turbulence. For the other two experiments the jet bore into the fluid with a mushroom-like head formed at its front. We observe, from Taylor's photographic evidence (see Fig. 8a), that for Re = 10 the jet penetrates only a few jet diameters before spreading out into a mushroom-like head. For higher Reynolds number fluids (Re = 200) the mushroom-like head is also observed (see Fig. 9a), but in this case the jet penetrates the fluid until the bottom of the box is reached.

To validate the technique presented in this paper we have simulated a jet impinging on a box of quiescent fluid so that a comparison with Taylor's results may be effected. We used a cylindrical box of diameter 12 cm and height 16 cm from which a colored jet falls vertically into the box at a prescribed velocity of 50 cm s^{-1} (see Fig. 5). The no-slip condition was applied on the box walls while on the axis of symmetry the symmetry condition (see [39]) was applied. The convergence criteria for the Poisson equation was EPS $= 10^{-6}$ and three particles per cell were employed to represent the free surface of the fluid. To simulate the incoming jet, a round nozzle of diameter $D = 4$ mm and 3 cm length was set 1 cm above the fluid surface in the box, from which the fluid jet travels at a constant velocity of 50 cm s^{-1}. Gravity was acting downward with $g_z = -1$ and the gravitational constant was taken to be $g = 981$ cm s^{-2}; this gave a Froude number of $\text{Fr} = \frac{U}{\sqrt{gD}} = 2.52409$. A mesh size of $\delta r = \delta z = 0.05$ cm was employed (120×400 cells within the mesh). Two runs were performed; the difference between these were only due to the value of viscosity. In the first run we set $\nu = 2.0$ $\text{cm}^2\,\text{s}^{-1}$ and for the second run we used $\nu = 0.1$ $\text{cm}^2\,\text{s}^{-1}$, giving

Reynolds numbers of

$$\mathrm{Re} = \frac{VD}{\nu} = 10, \quad \text{and} \quad \mathrm{Re} = \frac{VD}{\nu} = 200,$$

respectively. Figure 6 displays a series of snapshots of the jet flow simulation for Re = 10 at different times. Figure 7 displays the time evolution of the Re = 200 simulation. Figure 8a displays Taylor's photograph for the case of Re = 10 and Fig. 8b displays a front view of the three-dimensional jet shown in Fig. 8c. Figure 9a displays Taylor's photograph for the case of Re = 200 and Fig. 9b displays a front view of the three-dimensional jet shown

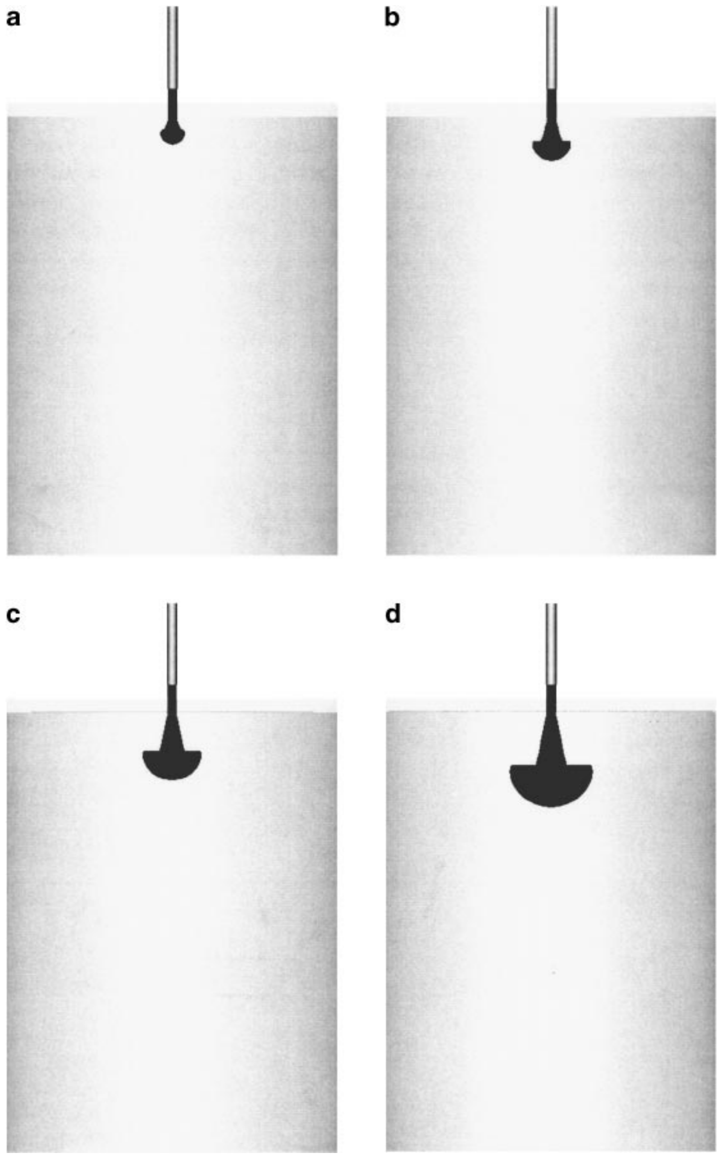


FIG. 6. Jet flow experiment: Re = 10. Fluid flow visualization at nondimensional times: (a) $t = 15.0$, (b) $t = 27.5$, (c) $t = 57.5$, (d) $t = 115.0$.

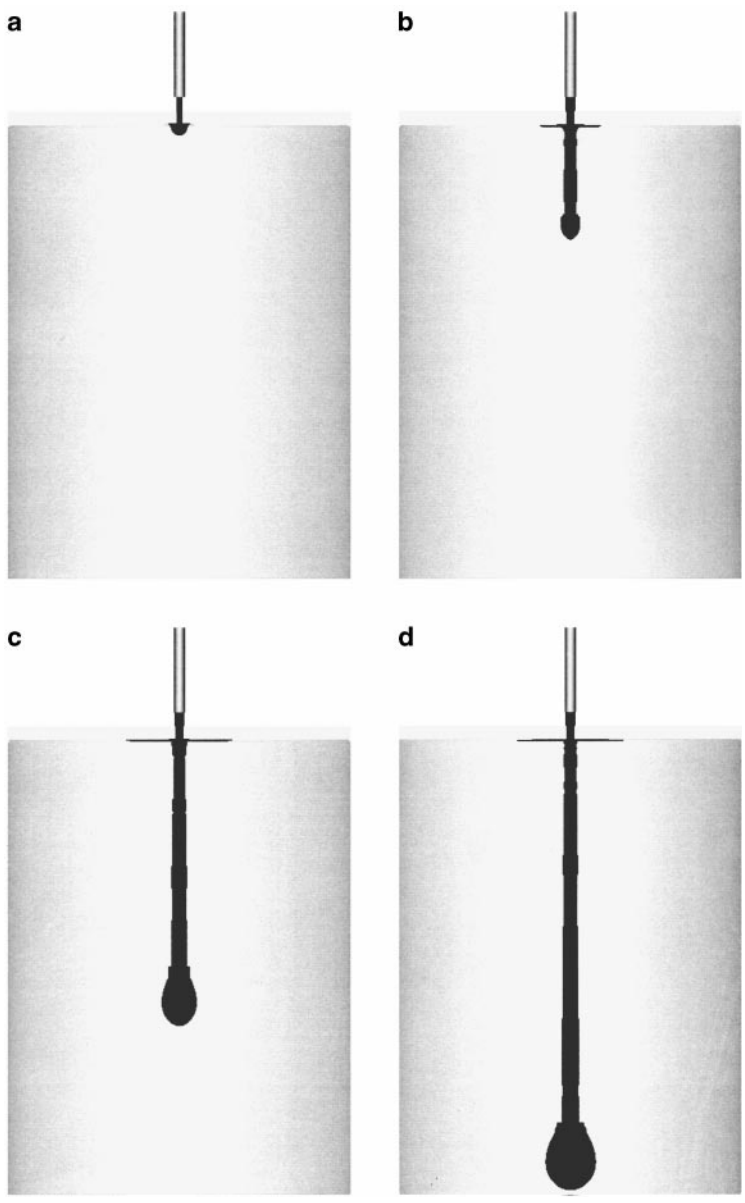


FIG. 7. Jet flow experiment: Re = 200. Fluid flow visualization at nondimensional times: (a) $t = 7.5$, (b) $t = 20.0$, (c) $t = 57.5$, (d) $t = 110.0$.

in Fig. 9c. As we can see the agreement with Taylors's experiment is excellent although for the Re = 200 case the simulation appears to underestimate the amount of diffusion. This demonstrates that the technique presented here can indeed cope with complicated axisymmetric flows.

7.2. *Splashing Drop Simulation*

We consider a pool of diameter 22 cm and height 7 cm full of a quiescent fluid. A spherical drop of fluid of diameter 3.2 cm was placed at a height of 3 cm above the pool with an

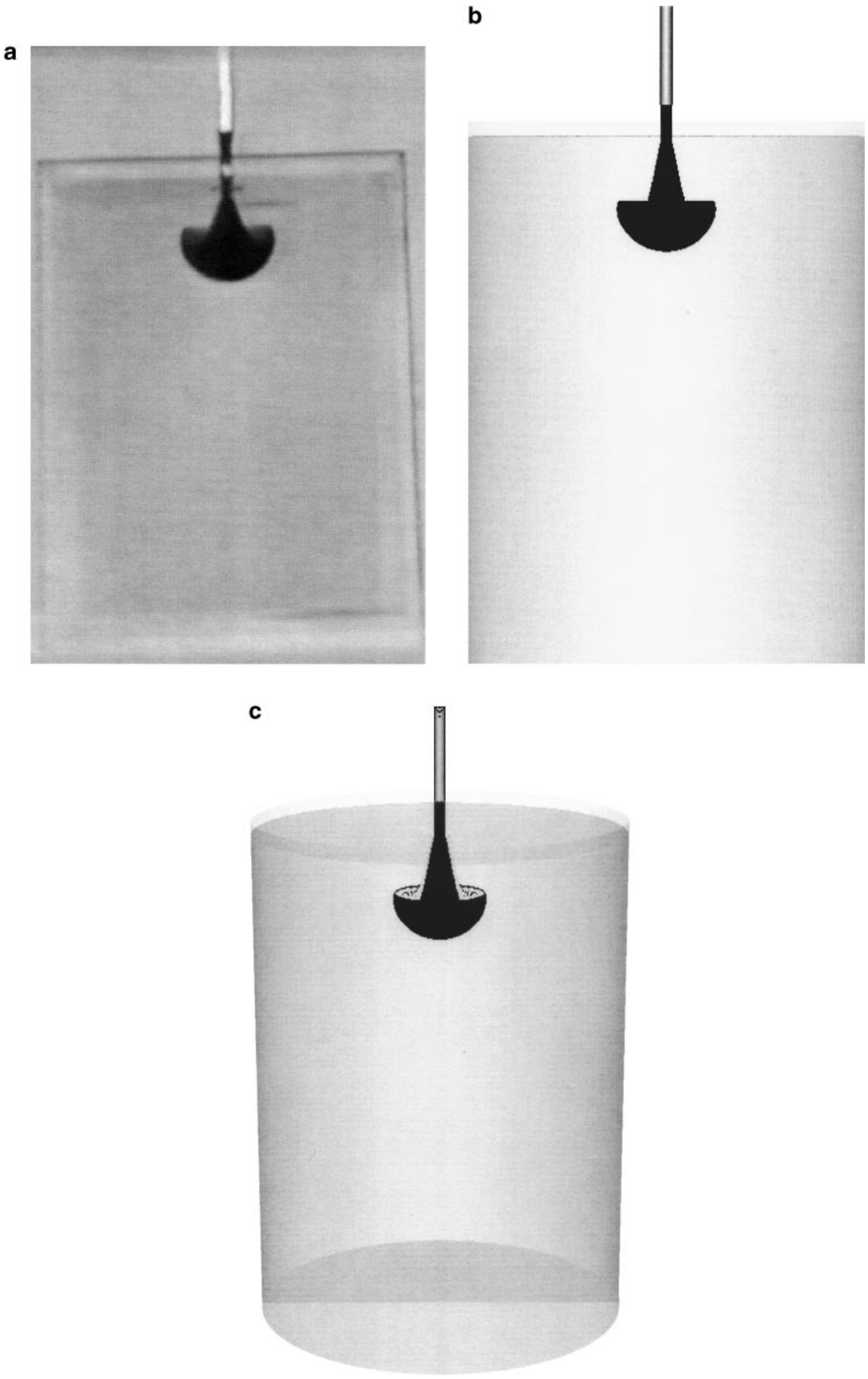


FIG. 8. Jet flow simulation: Re = 10. Fluid flow visualization at $t = 115.0$. Comparison with experimental data: (a) Taylor's photograph; (b) front view of the jet flow simulation; (c) three-dimensional view.

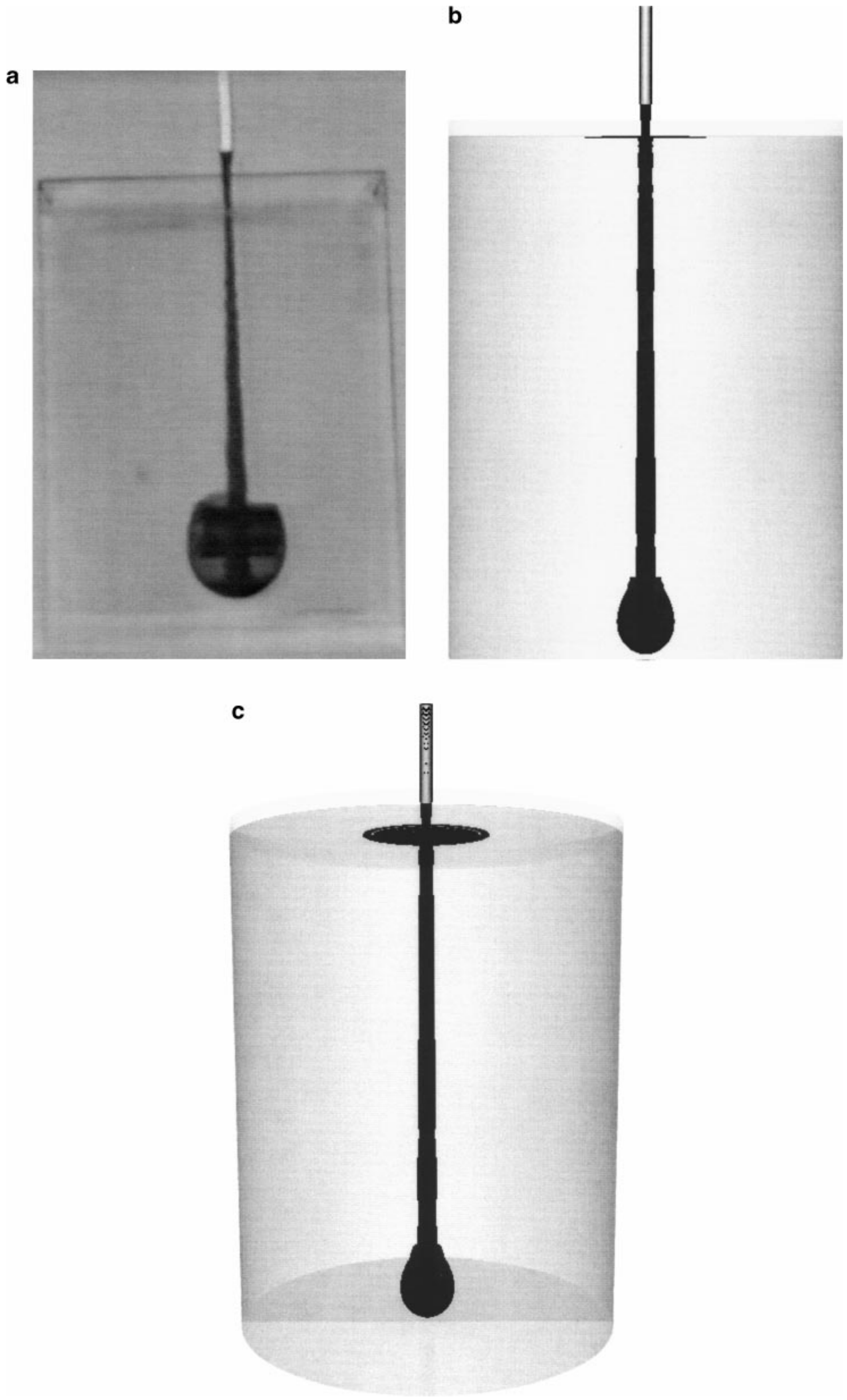


FIG. 9. Jet flow simulation: Re = 200. Fluid flow visualization at time $t = 110$. Comparison with experimental data: (a) Taylor's photograph; (b) front view; (c) three-dimensional view.

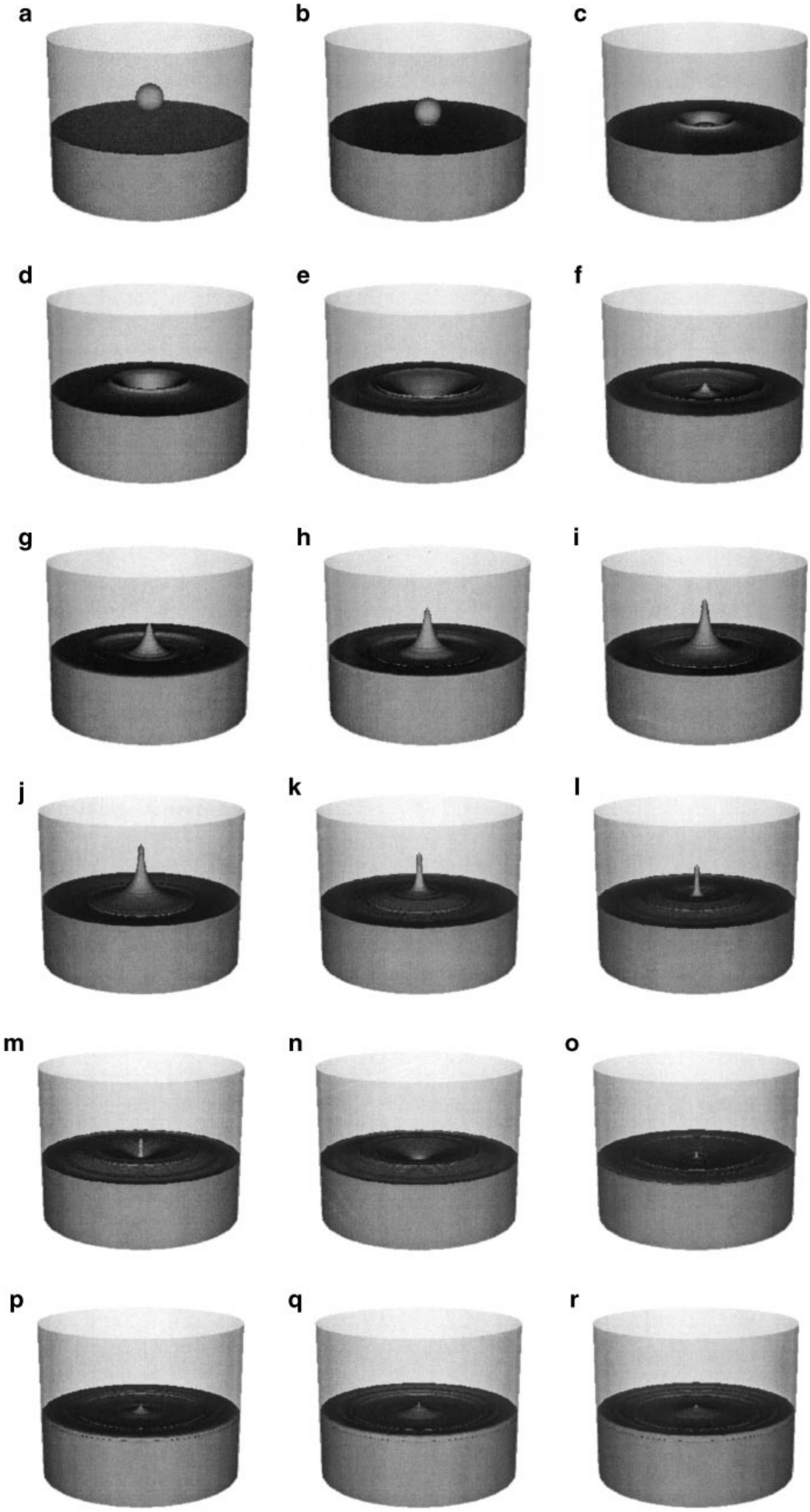


FIG. 10. The splashing of a drop of a Newtonian fluid: fluid flow visualization at different times.

initial velocity of -100 cm s^{-1}. It was then allowed to fall under gravity into the pool. The cell size employed was $\delta r = \delta z = 0.2$ cm; gravity was taken to be $g = -981$ cm s^{-2}; and viscosity was chosen to be $\nu = 3$ cm^2 s^{-1}. The scaling parameters were set to $U = 100$ cm s^{-1} and $L = 3.2$ cm, giving $\mathrm{Re} = 106.667$ and $\mathrm{Fr} = 1.7848$. A series of snapshots in time of the simulated splashing drop are displayed in Fig. 10. It is interesting to observe that, as the drop disappears within the bulk of the fluid, a thin jet emerges composed entirely of the original quiescent fluid. Further, a circular hydraulic jump can be observed moving away from the center.

7.3. *Simulation of Container Filling*

Many manufacturing industries are concerned with extruding their product into a container. Often the fluid product (e.g., paint, milk, or molten alloys) is non-Newtonian and GENSMAC has been demonstrated to deal with such fluids (see [40]). An extension to non-Newtonian fluids (e.g., generalized Newtonian fluids using the cross and power-law models) is not difficult and closely follows the discussion in [40]. Next we present the simulation of the filling of two popular containers which are commonly used by the industry.

Simulation of the filling of a circular tub. Here we shall simply consider the extrusion of a Newtonian fluid into a "circular tub" from a cylindrical nozzle (see Fig. 11a); The flow domain is assumed to be as shown in Fig. 11b. The code was run with the following input data:

$$\begin{aligned} U &= 100 \text{ cm s}^{-1} \text{ (nozzle velocity)} \\ D &= 1 \text{ cm (nozzle diameter)} \\ \nu &= 50 \text{ cm}^2 \text{ s}^{-1} \text{ (fluid kinematic viscosity).} \end{aligned}$$

This gave a Reynolds number of $\mathrm{Re} = UD/\nu = 2.0$ and a Froude number of $\mathrm{Fr} \approx 3.193$. The flow domain was defined by setting $L_1 = 5$ cm, $L_2 = 4$ cm, and $H = 6$ cm (see Fig. 11b). The nozzle was placed at a distance of 1 cm above the tub. A mesh spacing of $\delta r = \delta z = 0.1$ cm

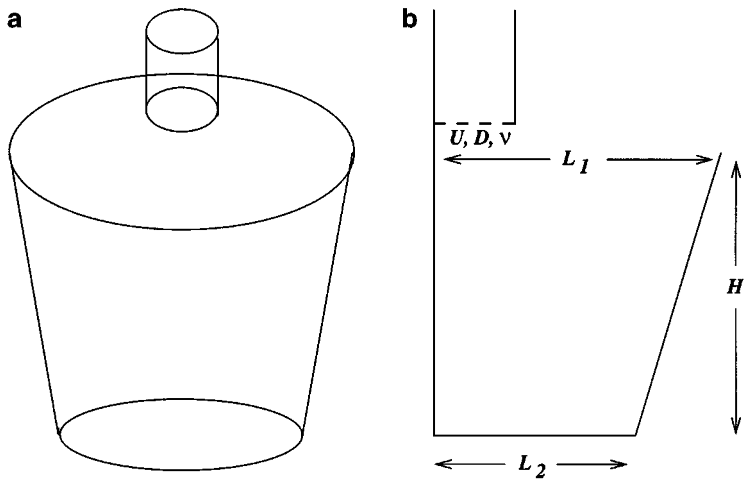


FIG. 11. The filling of a circular tub.

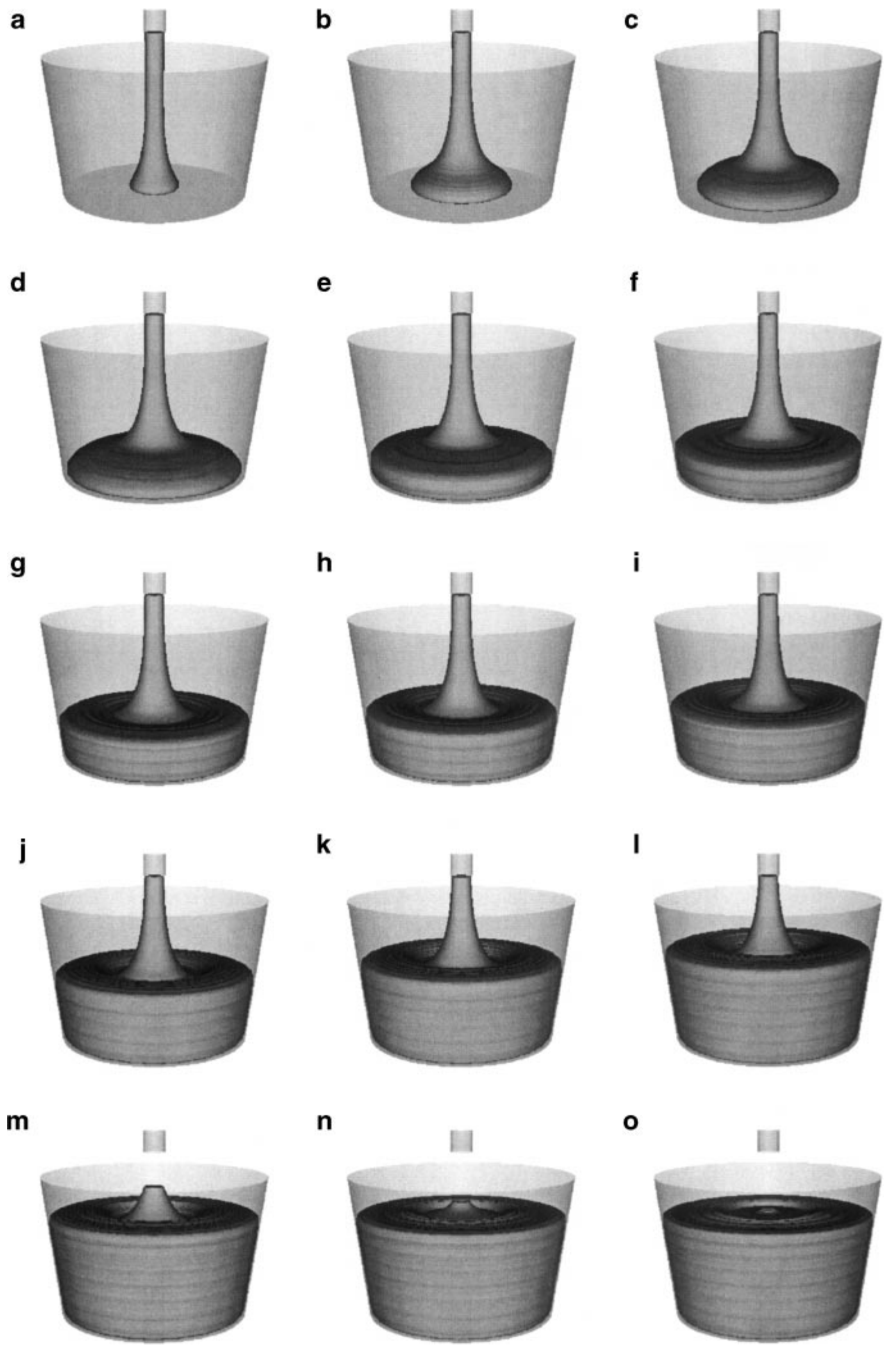


FIG. 12. Simulation of the filling of a circular tub: fluid flow visualization at different times.

was employed. The simulated fluid plots are displayed at a sequence of given times in Fig. 12. This fill represented a reasonably successful industrial fill: there was no spillage through splashing, spluttering, or sloshing although the final product may exhibit the phenomenon known in the trade as doming—that is, the surface of the final product may not be entirely horizontal and may slope down at the sides.

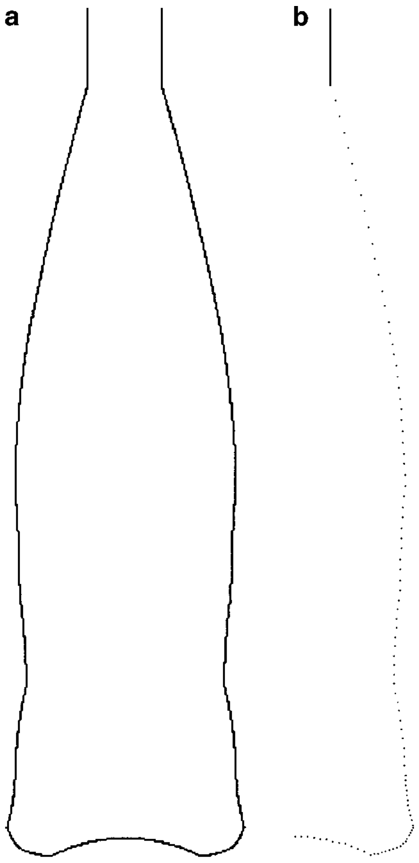


FIG. 13. The filling of a bottle.

Simulation of the filling of a bottle. To demonstrate that the code can cope with complex-shaped domains we simulated the filling of a curved bottle with a Newtonian fluid (see Fig. 13a). The following data were employed:

$$U = 100 \text{ cm s}^{-1} \text{ (nozzle velocity)}$$
$$D = 1.4 \text{ cm (nozzle diameter)}$$
$$\nu = 50 \text{ cm}^2 \text{ s}^{-1} \text{ (fluid kinematic viscosity).}$$

This gave a Reynolds number of $\mathrm{Re} = UD/\nu = 2.8$ and a Froude number of $\mathrm{Fr} \approx 2.6984$. The flow domain was defined by the set of points shown in Fig. 13b. The mesh spacing chosen was $\delta r = \delta z = 0.1$ cm (34×286 cells within the mesh), the convergence criteria for the Poisson equation was $EPS = 10^{-7}$, and three particles per cell were used to represent the fluid surface. A nozzle having 4 cm length was situated 1.5 cm above the bottle. Figure 14 displays the simulated fluid flow configuration at different times.

8. VALIDATION AND CONVERGENCE RESULTS

In this section we present quantitative evidence showing that the technique presented in this paper does converge to the solution of the underlying Eqs. (1)–(3) presented in Section 2.

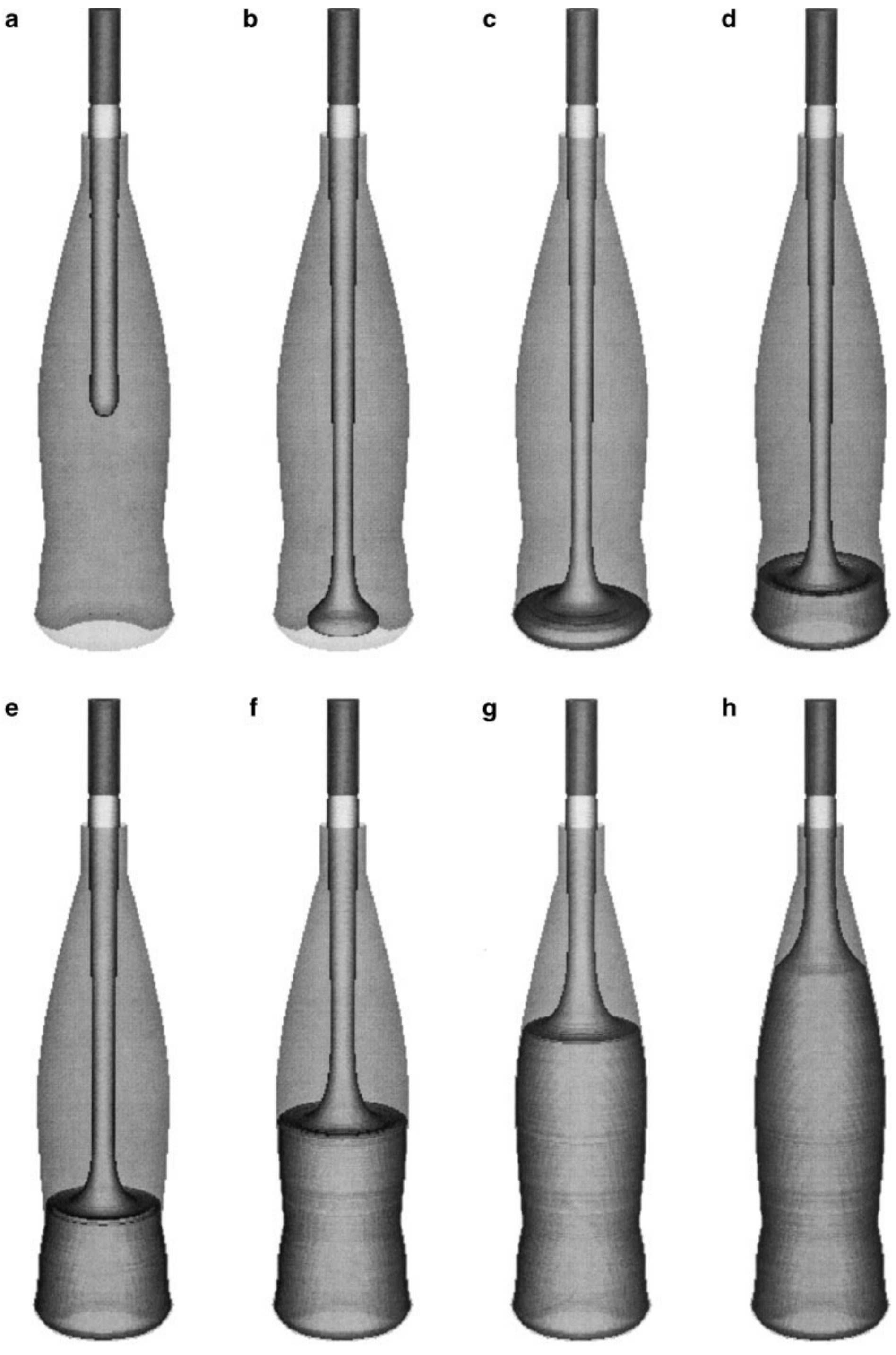


FIG. 14. Simulation of the filling of a bottle. Fluid flow visualization at different times: (a) $t = 10.715$, (b) $t = 17.150$, (c) $t = 32.150$, (d) $t = 50.000$, (e) $t = 85.715$, (f) $t = 135.715$, (g) $t = 189.286$, (h) $t = 212.159$.

8.1. *Comparison with Linear Theory for Standing Waves*

The problem we considered for validating the code consists of an axisymmetric irrotational flow (cylindrical pool) with a sufficiently small surface displacement. We have compared the computed and the analytical solutions, using linear theory for axisymmetric standing gravity waves. It is easy to show that the solution to this problem is composed of linear combinations of modes of the form (see Stoker [46])

$$u(r,z,t) = -Am\cos(\sigma t)J_1(mr)\cosh(mz) \tag{28}$$

$$v(r,z,t) = Am\cos(\sigma t)J_0(mr)\sinh(mz) \tag{29}$$

$$\xi(r,t) = \frac{A\sigma}{g}\sin(\sigma t)J_0(mr)\cosh(mH), \tag{30}$$

where g is the gravity and $\sigma = \sqrt{gm\tanh(mH)}$ is the angular frequency of the wave. H is the initial depth of the pool, R is the radius of the pool, m is a constant such that $J_1(mR) = 0$, A is a small arbitrary constant defining the wave amplitude, and ξ is the displacement of the free surface with respect to the initial surface.

The initial profile of the velocities and the surface elevation are obtained by taking $t = 0$ in (28)–(30). It is easily seen that the solution, defined by (28)–(30), satisfies the following boundary conditions (see Fig. 15):

$$\begin{cases} \frac{\partial v(r,z,t)}{\partial r}\Big|_{r=0} = 0 \\ u(0,z,t) = 0 \end{cases} \qquad \begin{cases} \frac{\partial v(r,z,t)}{\partial r}\Big|_{r=R} = 0 \\ u(R,z,t) = 0 \end{cases} \qquad \begin{cases} \frac{\partial u(r,z,t)}{\partial z}\Big|_{z=0} = 0 \\ v(r,0,t) = 0 \end{cases}$$

We solved the problem depicted in Fig. 15 for different input data and compared the numerical solutions to the exact solutions given by (28)–(30). Next we present the results of some representative simulations. The following data were employed:

- $\nu = 0.1$, $g = -1$, $H = 7.6251$,
- $L = U = 1$, $\delta r = \delta z = 0.0766342$.

The parameters m and R were varied; all other parameters were held fixed. Quantitative results, summarized in Table I, show reasonable agreement between the analytical and the calculated frequency of oscillation and the maximum amplitude of the wave elevation. Nonetheless, the differences are significant. These, we believe, are due to the fact that we are solving the full Navier–Stokes equations; the analytic expressions (28)–(30) are, of course,

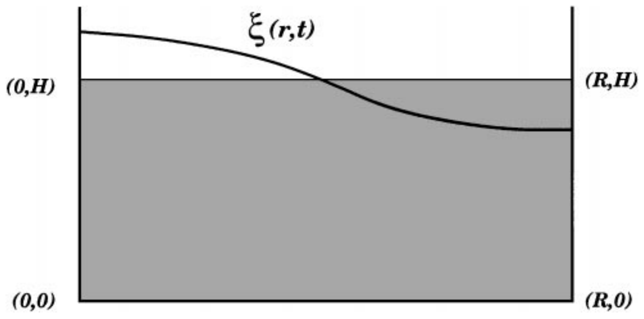


FIG. 15. Problem definition for the wave simulation.

TABLE I

m	R	σ_{exact}	σ_{calc}	%-error	$\xi_{\text{max}}^{\text{exact}}$	$\xi_{\text{max}}^{\text{calc}}$	%-error
1.00	3.8317	0.9905	0.9844	0.6158	0.3813	0.3408	10.610
0.50	7.6634	0.7000	0.7064	0.9116	0.3813	0.3646	4.3685
0.25	15.327	0.4844	0.4872	0.5881	0.3813	0.3722	2.3781

only valid for potential irrotational flow. Further evidence of this is provided in Table I since it can be seen that the differences decrease as the radius of the pool increases: the larger the wavelength the smaller will be the nonlinear and viscous effects since both the depth and the maximum amplitude of the wave were held fixed. For $m = 0.25$ and $R = 15.327$, Fig. 16 displays the time evolution of the surface elevation at $r = 0$ (i.e., $\xi(0, t) + H$). It can be seen that there is good agreement between the numerical and analytical results. Figure 17 shows the spatial variation of the surface elevation at time $t = \frac{\pi}{2\sigma}$. This corresponds to the first crest of the wave. Again, we can observe that there is good agreement between the two solutions.

8.2. *Mesh Refinement*

To demonstrate that the method of this paper does in fact converge as the mesh is refined we have performed three calculations with decreasing mesh spacings, both for the splashing drop and for the cavity filling problem.

For the splashing drop the following input data were employed:

- a cylindrical pool of quiescent fluid, diameter 22 cm and height 7 cm.

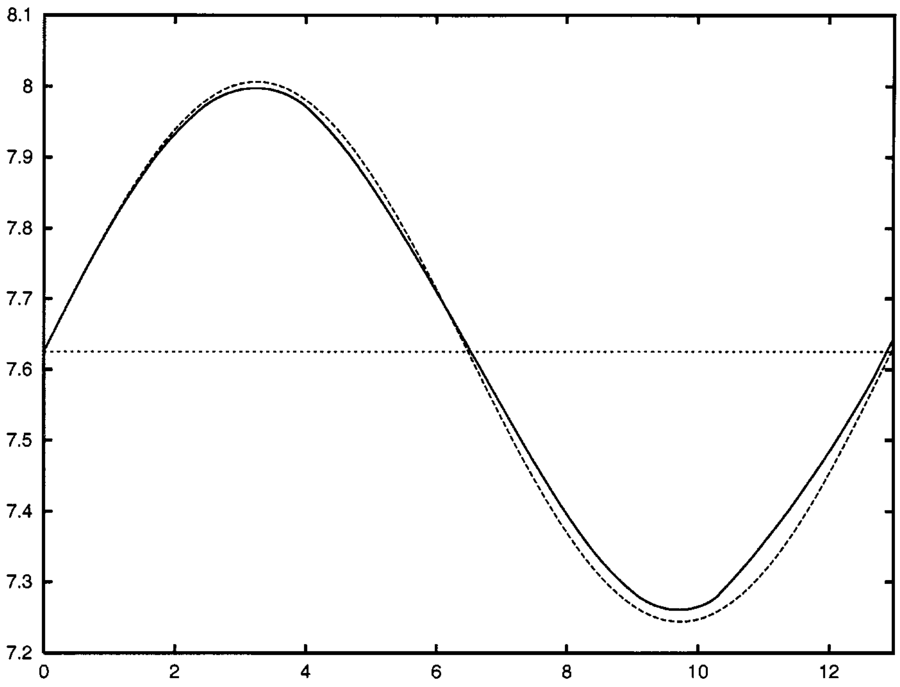


FIG. 16. Time evolution of the surface elevation at $r = 0$; — numerical solution, ---- analytical solution.

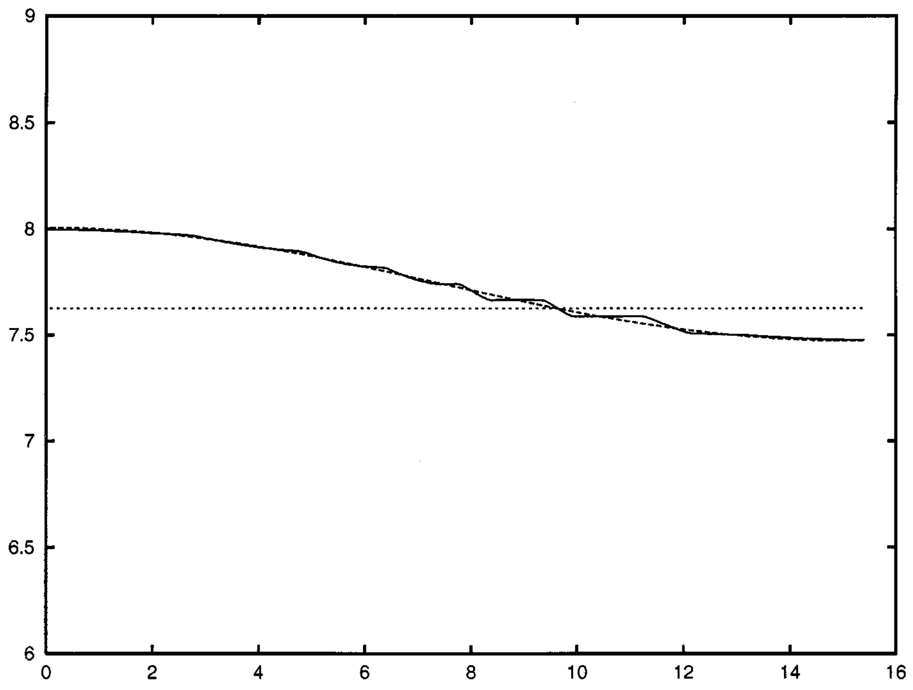


FIG. 17. Spatial variation of the surface elevation at $t = \frac{\pi}{2\sigma}$; — numerical solution, ---- analytical solution.

- a spherical drop of fluid of diameter 3.2 cm positioned initially 3 cm above the center of the pool with an initial velocity $V = -100$ cm s^{-1}.
- gravity assumed to be acting in the negative z-direction; i.e., $g_z = -981$ cm s^{-2}.
- kinematic viscosity $\nu = 3$ cm^2 s^{-1}.
- scaling parameters $L = 3.2$ cm (diameter of the drop), $U = 100$ cm s^{-1} (initial velocity of the drop), giving rise to Re = 106.67 and Fr = 1.7848.
- no-slip conditions applied on the side walls.
- Three particles per cell employed to represent the free surface of the fluid.

Figure 18 displays the configuration of the splashing drop simulation and its corresponding velocity field, calculated using the above data and three different mesh spacings. For the first run (see Fig. 18a) we employed a mesh spacing of $\delta r = \delta z = 0.4$ cm which gave a mesh size of 28×35, for the second run (see Fig. 18b) we used $\delta r = \delta z = 0.2$ cm, giving a mesh size of 56×70 and in the third run (see Fig. 18c) we employed a mesh spacing of $\delta r = \delta z = 0.1$ cm which produced a mesh size of 112×140.

We also considered the cavity filling problem presented in Section 7.3 and performed three runs to check the convergence of the finite difference solution as the mesh is refined. The following data were employed:

- a wedge-shaped circular container with dimensions $L_1 = 10$ cm, $L_2 = 4$ cm, and $H = 7$ cm (see Fig. 12b).
- nozzle diameter $D = 1.6$ cm.
- gravity was assumed to be acting in the negative z-direction; i.e., $g_z = -981$ cm s^{-2}.
- kinematic viscosity $\nu = 50$ cm^2 s^{-1}.
- scaling parameters $L = 1.6$ cm (diameter of the nozzle), $U = 100$ cm s^{-1} (fluid velocity at the nozzle) giving rise to Re = 3.2 and Fr = 2.52409.

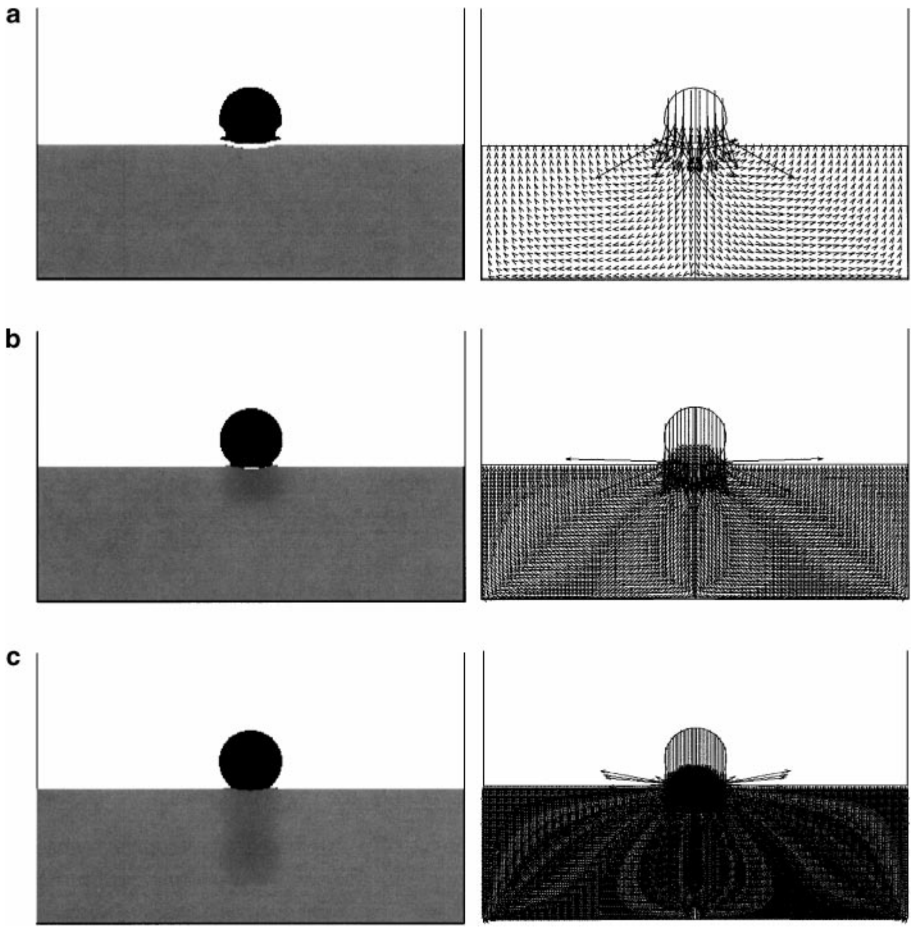


FIG. 18. Simulation of splashing drop on different grids. (a) grid 28 × 35; (b) grid 56 × 70; (c) grid 112 × 140. Fluid flow visualization and velocity plot at $t = 0.46875$.

- three particles per cell on the free surface.
- no-slip conditions applied on the container walls.

It should be stressed that the only difference between runs, in both cases, were the mesh spacings.

Figure 19 displays the fluid flow configuration of the jet filling and its corresponding velocity field at the nondimensional time of $t = 59.375$ on the three meshes. For the first run (see Fig. 19a) we employed a mesh spacing of $\delta r = \delta z = 0.4$ cm which gave rise to a 13×23 grid, for the second run (see Fig. 19b) we used a $\delta r = \delta z = 0.2$ which produced a 26×46 grid, while for the third run (see Fig. 19c) the mesh spacings were chosen to be $\delta r = \delta z = 0.1$ resulting in a mesh size of 52×92 cells.

In both cases the figures suggest that convergence is being obtained.

8.3. *Convergence Study*

The results of the previous section show that the method converges as the mesh is refined. To obtain an estimate of the rate of convergence we performed a number of runs

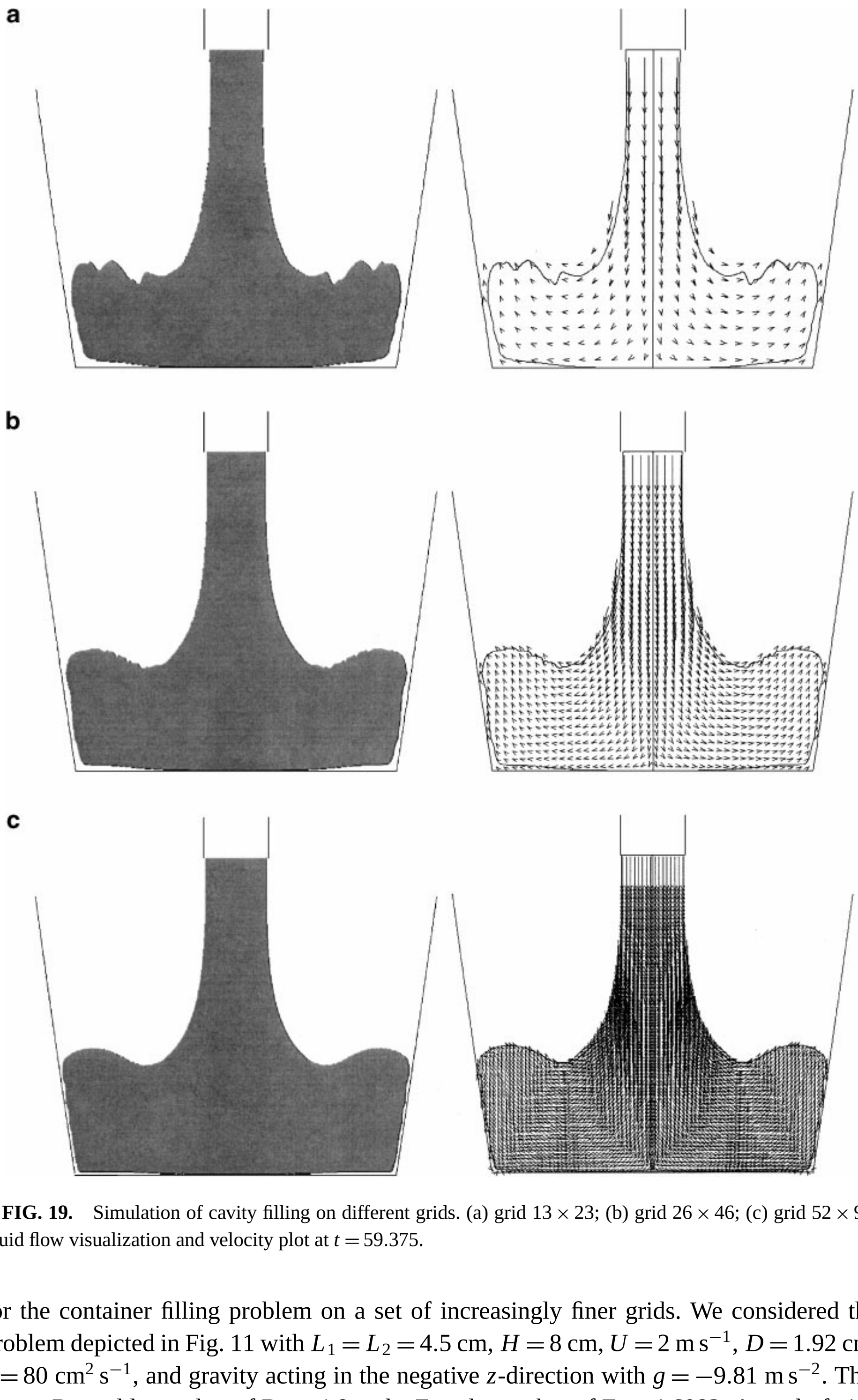


FIG. 19. Simulation of cavity filling on different grids. (a) grid 13×23; (b) grid 26×46; (c) grid 52×92. Fluid flow visualization and velocity plot at $t = 59.375$.

for the container filling problem on a set of increasingly finer grids. We considered the problem depicted in Fig. 11 with $L_1 = L_2 = 4.5$ cm, $H = 8$ cm, $U = 2\ \mathrm{m\,s^{-1}}$, $D = 1.92$ cm, $\nu = 80\ \mathrm{cm^2\,s^{-1}}$, and gravity acting in the negative z-direction with $g = -9.81\ \mathrm{m\,s^{-2}}$. This gave a Reynolds number of $\mathrm{Re} = 4.8$ and a Froude number of $\mathrm{Fr} \approx 4.6083$. A total of nine runs were performed on the following grids: Mesh I ($h = 0.0032$), Mesh II ($h = 0.0024$), Mesh III ($h = 0.0016$), Mesh IV ($h = 0.0012$), Mesh V ($h = 0.008$), Mesh VI ($h = 0.0006$),

TABLE II
***l_2-norm* of the Errors in the Velocity Field and Pressure for Different Mesh Spacings**

h	$\|\|U - U_h\|\|_2$	$\|\|V - V_h\|\|_2$	$\|\|P - P_h\|\|_2$
0.0032	0.0255	0.0893	0.4689
0.0024	0.0253	0.0556	0.4807
0.0016	0.0222	0.0486	0.1826
0.0012	0.0218	0.0423	0.1580
0.0008	0.0142	0.0264	0.1076
0.0006	0.0097	0.0180	0.1243
0.0004	0.0053	0.0092	0.0621
0.0003	0.0026	0.0045	0.0510

Mesh VII ($h = 0.0004$), Mesh VIII ($h = 0.0003$), and Mesh IX ($h = 0.0002$). As an analytic solution for this problem is not known we supposed that the solution on the finest mesh (Mesh IX) is the exact solution and computed the l_2-*norm* of the errors between this "exact solution" and those on the coarser grids. Table II displays the l_2-*norm* of the errors for the velocity components and pressure. These errors were calculated on the coarsest grid (Mesh I) by considering only full cells using the Matlab routine Interp2 to interpolate from the finer grids. The reason for considering the full cells only was that there were points near the boundaries on the coarsest grid which were not defined on the finer grids so that a fair comparison could not have been properly made.

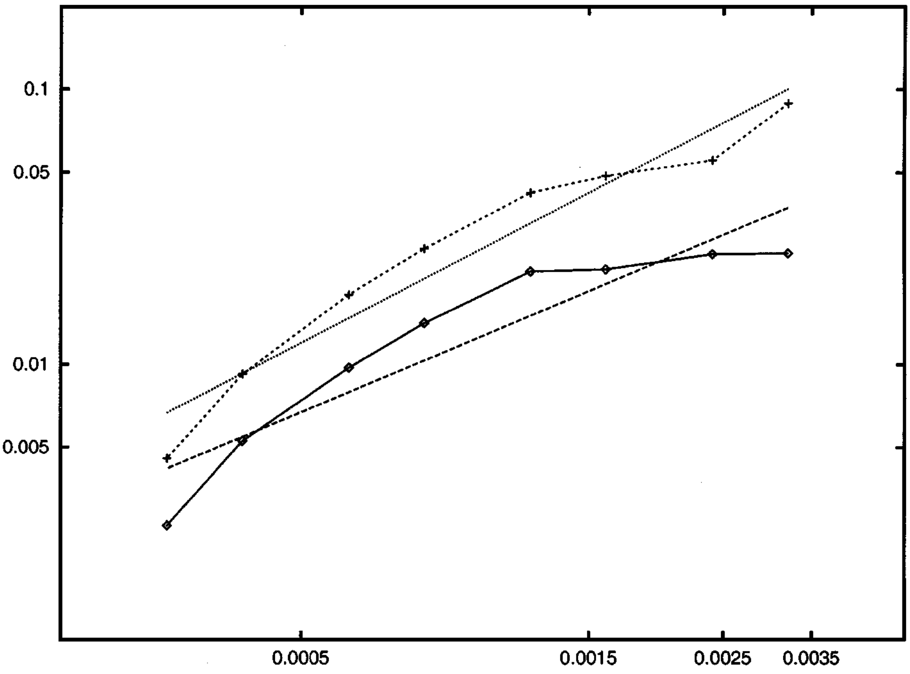


FIG. 20. Convergence rate study of the velocity components: $\diamond$ error in U_h, --- best fit ($0.7584 \times 10^1\ h^{0.925}$), $+$ error in V_h, $\cdots$ best fit ($0.7327 \times 10^2\ h^{1.14771}$).

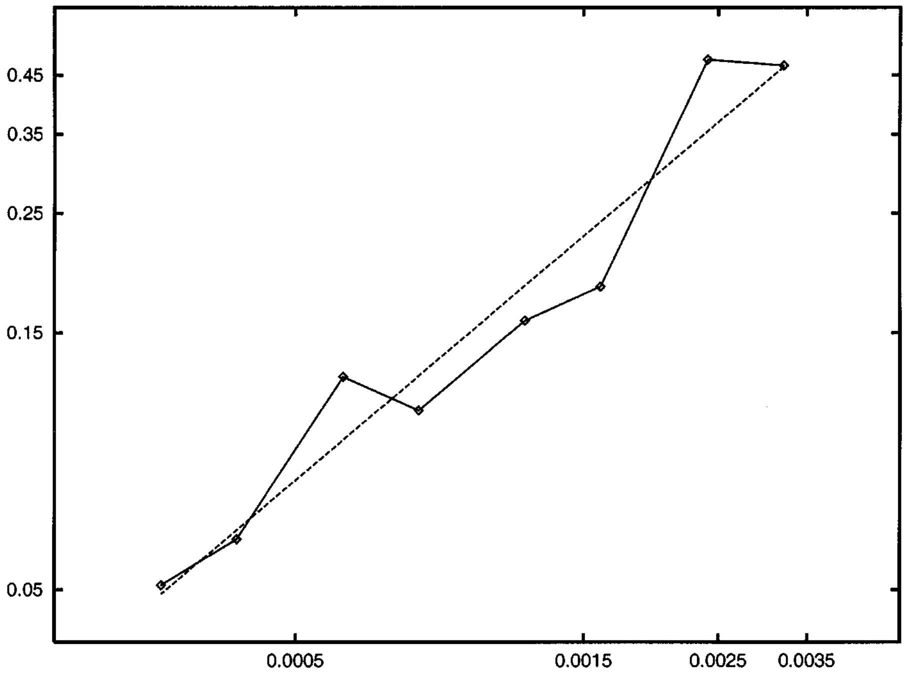


FIG. 21. Convergence rate study of the pressure field: ◇ error in P_h, - - - best fit ($0.1100 \times 10^3\, h^{0.9511}$).

As we can see from Table II, the errors in the velocity components are monotonically decreasing while the pressure displays an oscillatory behavior. We believe that this oscillatory behavior in the pressure, particularly for $h = 0.0006$, can be attributed to the fact that the boundaries do not fit the mesh lines exactly. In order to obtain an estimate of the rate of convergence a least square line is fitted to the log–log data in Table II; the convergence rate is then given by the slope of the fitted line. Figure 20 displays the log–log plot of the l_2-*norm* error of the velocity components and Fig. 21 displays the l_2-*norm* error for the pressure. The slope for the u-error is 0.925, for the v-error it is 1.15, while for the pressure the slope is 1.020. These convergence estimates are necessarily only approximate. Furthermore, if we neglect the less accurate solutions and start from $h = 0.0016$ then higher convergence estimates may be obtained for the velocity components. Indeed, the convergence rate appears then to be closer to 1.5 for both u and v. In conclusion, numerical convergence estimates have been obtained and the results show that the method presented in this paper is convergent of at least $O(h)$.

9. CONCLUSIONS

The GENSMAC code has been extended to cope with axisymmetric flows. The computational efficiency of the original code has been greatly increased with the use of surface marker particles only. A graphic interface permits easy data input while the ideas of solid modeling are employed to provide an output facility in the form of three-dimensional visualization. Several examples have been included, including a comparison with G. I. Taylor's experiment of a jet impinging onto a quiescent fluid. Finally, a comparison with standing waves and numerical convergence estimates were also provided.

ACKNOWLEDGMENTS

We thank Dr. Saulo M. Barros of the Department of Applied Mathematics of IME (Institute of Mathematics and Statistics) of the University of São Paulo for useful discussions and the referees for their constructive comments. We also thank the staff of the LCAD-USP de São Carlos (high performance computing laboratory) for their support on this project. This work has been funded by CNPq, Conselho Nacional de Pesquisa, and FAPESP, Fundação de Amparo a Pesquisa do Estado de São Paulo.

REFERENCES

1. W. Shyy, H. S. Vdaykumar, M. M. Rao, and R. W. Smith, *Computational Fluid Dynamics with Moving Boundaries* (Hemisphere, Washington, DC, 1996).
2. R. J. LeVeque, *Numerical Methods for Conservation Laws* (Verlag-Birkhauser, 1992).
3. C. Hirsch, Numerical computation of internal and external flows, in *Numerical Methods in Engineering* (Wiley, New York, 1988), Vols. 1 and 2.
4. R. D. Richtmyer and K. W. Morton, *Difference Methods for Initial Value Problems* (Interscience, New York, 1967).
5. J. Glimm, J. Grove, B. Lindquist, O. McBryan, and G. Tryggvason, The bifurcation of tracked scalar waves, *SIAM J. Sci. Stat. Comput.* **9**, 61 (1988).
6. I.-L. Chern, J. Glimm, O. McBryan, B. Plohr, and S. Yaniv, Front tracking for gas dynamics, *J. Comput. Phys.* **62**, 83 (1986).
7. P. Daripa, J. Glimm, B. Lindquist, M. Maesumi, and O. McBryan, On the simulation of heterogeneous petroleum reservoirs, in *Numerical Simulation in Oil Recovery*, edited by M. Wheeler (Springer-Verlag, New York, 1988).
8. G. Moretti, Computation of flows with shocks, *Ann. Rev. Fluid Mech.* **19**, 313 (1987).
9. C. S. Peskin, Numerical analysis of blood flow in the heart, *J. Comput. Phys.* **25**, 220 (1977).
10. L. J. Fauci and C. S. Peskin, A computational model of aquatic animal locomotion, *J. Comput. Phys.* **77**, 85 (1988).
11. A. L. Fogelson and C. S. Peskin, A Fast numerical method for solving the three-dimensional Stokes equations in the presence of gas suspended particles, *J. Comput. Phys.* **79**, 50 (1988).
12. G. R. Baker and D. W. Moore, The rise and distorsion of a two-dimensional gas bubble in an inviscid liquid, *Phys. Fluids A1* **9**, 1451 (1989).
13. T. M. Tsai and M. J. Miksis, Dynamics of a drop in a constricted capillary tube, *J. Fluid Mech.* **274**, 197 (1994).
14. J. C. S. Meng and J. A. L. Thomson, Numerical studies of some nonlinear hydrodynamic problems by discrete vortex element methods, *J. Fluid Mech.* **84**, 433 (1978).
15. G. Tryggvason, Numerical simulations of the Rayleigh–Taylor instability, *J. Comput. Phys.* **75**, 253 (1988).
16. S. O. Unverdi and G. Tryggvason, A front-tracking method for viscous, incompressible, multi-fluid flows, *J. Comput. Phys.* **100**, 25 (1992).
17. S. Osher and J. A. Sethian, Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton–Jacobi formulations, *J. Comput. Phys.* **79**, 1 (1988).
18. M. Sussman, P. Smereka, and S. Osher, A level set approach for computing solutions to incompressible two-phase flow, *J. Comput. Phys.* **114**, 146 (1994).
19. J. B. Bell and D. L. Marcus, A second-order projection method for variable-density flows, *J. Comput. Phys.* **101**, 334 (1992).
20. F. Beaux and S. Banerjee, Numerical simulation of three-dimensional two-phase flows by means of a lever set method, in *ECCOMAS 96* (Wiley, New York, 1996).
21. F. Harlow and J. E. Welch, Numerical calculation of time-dependent viscous incompressible flow of fluid with a free surface, *Phys. Fluids* **8**, 2182 (1965).
22. A. A. Amsden and F. H. Harlow, The SMAC method: A numerical technique for calculating incompressible fluid flow, Report LA-4370 (Los Alamos Scientific Laboratory, 1971).

23. H. Miyata, Finite difference similation of breaking waves, *J. Comput. Phys.* **65**, 179 (1986).
24. J. A. Viecelli, A computing method for incompressible flows bounded by moving walls, *J. Comput. Phys.* **8**, 119 (1971).
25. C. W. Hirt and J. P. Shannon, Free surface stress conditions for incompressible flow calculations, *J. Comput. Phys.* **2**, 403 (1968).
26. H. S. Udaykumar, H.-C. Kan, W. Shyy, and R. Tran-Son-Tay, Multiphase dynamics in arbitrary geometries on fixed Cartesian grids, *J. Comput. Phys.* **137**, 366 (1997).
27. W. F. Noh and P. R. Woodward, SLIC (Simple Line Interface Calculation) *Lecture Notes in Physics*, edited by A. I. van der Vooren and P. J. Zandbergen (Springer-Verlag, New York/Berlin, 1976), Vol. 59.
28. A. J. Chorin, Flame advection and propagation algorithms, *J. Comput. Phys.* **35**, 1 (1980).
29. A. F. Ghoniem, A. J. Chorin, and A. K. Oppenheim, Numerical modeling of turbulent flow in a combustion tunnel, *Phil. Trans. Roy. Soc. London A* **304**, 303 (1982).
30. J. A. Sethian, Turbulent combustion in open and closed vessels, *J. Comput. Phys.* **54**, 425 (1984).
31. C. W. Hirt and B. D. Nichols, Volume of fluid (VOF) method for the dynamics of free boundaries, *J. Comput. Phys.* **39**, 201 (1981).
32. B. D. Nichols, C. W. Hirt, and R. S. Hotchkins, *SOLA-VOLF: A solution algorithm for transient fluid flow with multiple free boundaries*, Technical Report LA-8355 (Los Alamos National Laboratory, Aug. 1988).
33. M. D. Torrey, L. D. Cloutman, R. C. Mjolsness, and C. W. Hirt, *NASA-VOF2D: A computer program for incompressible flow with free surfaces*, Technical Report LA-10612-MS (Los Alamos National Laboratory, Dec. 1985).
34. M. D. Torrey, R. C. Mjolsness, and L. R. Stein, *NASA-VOF3D: A three-dimensional computer program for incompressible flows with free surface*, Technical Report LA-11009-MS (Los Alamos National Laboratory, July 1987).
35. D. B. Kothe and R. C. Mjolsness, RIPPLE: A new model for incompressible flows with free surfaces, *AIAA J.* **30**, 2694 (1992).
36. 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).
37. C. W. Hirt, *Flow-3D Users Manual* (Flow Sciences, Los Alamos, 1988).
38. E. G. Puckett, A. S. Almgren, J. B. Bell, D. L. Marcus, and W. J. Rider, A high order projection method for tracking fluid interfaces in variable density incompressible flows, *J. Comput. Phys.* **130**, 269 (1997).
39. M. F. Tome and S. McKee, GENSMAC: A computational marker-and-cell method for free surface flows in general domains, *J. Comput. Phys.* **110**, 171 (1994).
40. M. F. Tome, B. Duffy, and S. McKee, A numerical technique for solving unsteady non-Newtonian free surface flows, *J. Non-Newtonian Fluid Mech.* **62**, 9 (1996).
41. M. F. Tome, GENSMAC: *A Multiple Free Surface Fluid Flow Solver*, Ph.D. Thesis (University of Strathclyde, Glasgow, U.K., 1993).
42. G. K. Batchelor, *An Introduction to Fluid Dynamics* (Cambridge University Press, Cambridge, 1967).
43. G. I. Taylor, *Film Notes for Low-Reynolds Number Flows* (Cambridge University Press, Cambridge, UK, 1967).
44. W. Schroeder, K. Martin, and B. Lorensen, *The Visualization Toolkit—An Object-Oriented Approach to 3D Graphics* (Prentice-Hall, Englewood Cliffs, NJ, 1996).
45. R. Minghim and M. C. F. Oliveira, *Three-dimensional visualization of free surface flows*, Internal Report 10-ICMSC-USP (São Paulo, Brazil, 1996). [Unpublished]
46. J. J. Stoker, *Water Waves* (Interscience, New York, 1957).