The stream volume of fluid advection algorithm

Dalton J.E. Harvie* David F. Fletcher*

(Received 7 August 2000)

Abstract

The Volume of Fluid (VOF) method is a powerful tool for modelling the movement of free surface fluid flows. In this paper, a new VOF advection algorithm is presented, termed the Stream scheme. The algorithm uses a linear piecewise method for free surface reconstruction, coupled to a unique fully multidimensional method of cell boundary flux integration. Comparisons with other VOF advection algorithms show the performance of the new scheme to be good.

*Department of Chemical Engineering, University of Sydney, NSW, 2006, AUSTRALIA.
mailto:daltonh@mech.eng.usyd.edu.au and mailto:davidf@chem.eng.usyd.edu.au

[0]See http://anziamj.austms.org.au/V42/CTAC99/Harv for this article and ancillary services, Published 27 Nov 2000.

Contents

1 Introduction

The Volume of Fluid (VOF) method is a convenient and powerful tool for modelling fluid flows which contain a free surface [3]. Under the VOF method, fluid location is recorded using a volume of fluid function. In a single fluid

calculation, this function is defined as unity within fluid regions, and zero elsewhere. In numerical fluid simulations, where the VOF function is averaged over each computational cell, the function becomes one in cells containing only fluid, zero in cells containing no fluid, and between these values in cells which contain a free surface. The method used to advect the VOF function is the subject of this work.

In order to preserve the discrete nature of the fluid free surface interfaces, current VOF advection algorithms employ a two stage process. Firstly, free surface interfaces are 'reconstructed' from the VOF data, so that a geometrical profile is found which approximates the actual free surface location. Changes in VOF values are then calculated by integrating fluid fluxes over cell boundaries, using the geometrical profile to indicate the location of fluid regions.

Excellent reviews of past and present VOF advection methods have been given by Rider and Kothe [6] and Rudman [7], so only a brief overview of some of the methods available will be given here. The different advection methods can be loosely classified according to the technique used to reconstruct the free surfaces in each cell, and by the method used to perform the boundary flux integrations [6].

VOF advection methods that represent free surface interfaces as lines directed parallel to one of the grid coordinates are known as piecewise constant schemes. The Simple Line Interface Calculation (SLIC) [4] is an early example of a piecewise constant scheme. The Hirt-Nichols (H-N) scheme, as used

in the SOLA-VOF code [3], is a variation of the piecewise constant method. Under the H-N scheme, free surface interfaces are orientated primarily in directions parallel to grid coordinates, but are allowed the greater freedom of a stair-shaped profile if the local distribution of the VOF function permits.

The alternative to representing free surface interfaces as lines parallel to one of the grid coordinates is to orientate free surface interfaces in a direction perpendicular to the locally evaluated gradient of the VOF function. VOF advection schemes using this method of free surface reconstruction are known as piecewise linear schemes, and have been shown to be significantly more accurate than piecewise constant schemes [7, 6, 1]. The Stream algorithm detailed in this study is a piecewise linear scheme.

The method of integration used to determine cell boundary fluxes is also used to classify VOF advection techniques. Under operator split schemes, boundary fluxes are calculated independently in each coordinate direction, often with some type of limiter employed to reduce possible undershoots or overshoots occurring in cell averaged values of the VOF function. The Youngs algorithm is an example of an operator split scheme [8].

Multidimensional schemes can be more accurate and efficient in calculating cell boundary fluxes than operator split schemes [6]. Under a multidimensional scheme, cell boundary fluxes are calculated with a dependence between fluxes calculated in each of the coordinate directions. The Stream scheme, as developed in this study, is a fully multidimensional scheme.

In this paper we detail the basic theory behind the Stream algorithm,

and analyse the performance of the algorithm when simulating a fluid body deforming in a single vortex. More specific details of the theory and implementation of the Stream scheme can be found in Harvie and Fletcher [2].

2 The Stream VOF Advection Algorithm

2.1 The Basic Method

The idea behind the Stream algorithm is simple, and can be summarised as follows:

1. Fluid interfaces are reconstructed in each cell using a piecewise linear interface method.
2. A semi-continuous velocity field is defined throughout the computational region, based on the staggered cell boundary velocities.
3. Donating regions for each cell boundary are defined by integrating back in time for the duration of the computational time step along fluid streamlines passing through the examined boundary.
4. Boundary fluxes are calculated as the intersection between each donating region and all fluid locations.

The primary tasks involved in implementing the Stream algorithm are defining a suitable velocity field, reconstructing the fluid free surfaces, and integrating along streamlines to determine fluid volume fluxes occurring over each boundary. It is these three topics which we now examine.

2.2 Defining the Velocity Field

The velocity at any point in a given cell i, j is defined as

$$\mathbf{V}(x, y) = \{\chi_b x + \chi_y\}\,\mathbf{i} + \{-\chi_b y - \chi_x\}\,\mathbf{j}, \tag{1}$$

where

$$\chi_b = \frac{u_{i+\frac{1}{2}} - u_{i-\frac{1}{2}}}{x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}}} = -\frac{v_{j+\frac{1}{2}} - v_{j-\frac{1}{2}}}{y_{j+\frac{1}{2}} - y_{j-\frac{1}{2}}}\,, \tag{2}$$

$$\chi_x = -\chi_b y_{j-\frac{1}{2}} - v_{j-\frac{1}{2}} \quad \text{and} \quad \chi_y = -\chi_b x_{i-\frac{1}{2}} + u_{i-\frac{1}{2}}\,. \tag{3}$$

Integer subscripts in these equations refer to cell centred quantities, while integer values plus or minus a half refer to quantities located at cell upper or lower boundaries, respectively. The velocity in the x coordinate direction is u, and that in the y direction is v. Note that the equality in equation (2) follows from the continuity equation applied to the examined cell.

Equation (1) specifies a velocity field which is semi-continuous—the field is discontinuous at cell vertices, and the tangential component of the velocity

is discontinuous across cell boundaries. Such discontinuities would be problematic if we were to integrate along streamlines from cell vertices, but the integration method we present later avoids this inconvenience. Note that fluid fluxes over cell boundaries are constant along the length of each boundary. Thus, streamlines intersecting each boundary, which are separated by equal volume fluxes, are equally spaced.

2.3 Free Surface Interface Reconstruction Using the Error Minimisation Method

A piecewise linear interface method is used by the Stream scheme for locating fluid regions. The method involves two steps for each interface reconstruction; determining the gradient of the interface within each cell, and positioning the interface within each cell to equate the calculated cell averaged VOF volume to the volume contained between the free surface and cell boundaries.

The error minimisation gradient calculation method presented here is based on the concept pioneered by Pilliod [5], however, a significant change to the scheme has been made. As the name implies, under the error minimisation scheme each interface gradient has associated with it an error function. When this error function is minimised, we find the optimal free surface gradient within the cell.

The error function is defined as follows.

1. Given a gradient, the interface is reconstructed such that the volume of fluid contained between the interface and cell boundaries is equal to the volume of fluid within the examined cell.

2. The fluid interface is continued beyond the boundaries of the examined cell, so that it traverses a total of $3 \times 3 = 9$ cells, with the examined cell at the centre.

3. Volume of fluid functions, F^*, are calculated for each of the surrounding 8 cells based on the extended reconstructed interface.

4. Finally, the error associated with the reconstructed interface is calculated. Under the Pilliod scheme the error function is defined as

$$E_{\text{Pilliod } i,j} = \sum_{\substack{n=i-1,i+1 \\ m=j-1,j+1}} \left(F_{n,m} - F^*_{n,m}\right)^2, \tag{4}$$

where $F_{n,m}$ is the actual VOF function for cell n, m and $F^*_{n,m}$ is the VOF function for cell n, m based on the extended reconstructed interface, as described above. In this work, the error function is defined as

$$E_{i,j} = \frac{E_{\text{void}}}{E_{\text{void,max}}} + \frac{E_{\text{fluid}}}{E_{\text{fluid,max}}}, \tag{5}$$

where

$$E_{\text{void}} = \sum_{\substack{n=i-1,i+1 \\ m=j-1,j+1}} \left\{\max\left(F_{n,m} - F^*_{n,m}, 0\right) \times A_{n,m}\right\}^3, \tag{6}$$

$$E_{\text{void,max}} = \sum_{\substack{n=i-1,i+1 \\ m=j-1,j+1}} \left\{ \left(1 - F^*_{n,m}\right) \times A_{n,m} \right\}^3, \tag{7}$$

$$E_{\text{fluid}} = \sum_{\substack{n=i-1,i+1 \\ m=j-1,j+1}} \left\{ \max\left(F^*_{n,m} - F_{n,m}, 0\right) \times A_{n,m} \right\}^3 \tag{8}$$

and

$$E_{\text{fluid,max}} = \sum_{\substack{n=i-1,i+1 \\ m=j-1,j+1}} \left\{ F^*_{n,m} \times A_{n,m} \right\}^3. \tag{9}$$

The error function defined here differs from the Pilliod function in two significant ways. Firstly, the Stream error function comprises two separate errors, one resulting from the fluid region adjacent to the reconstructed interface and the other from the void region adjacent to the reconstructed interface. As these errors are normalised separately against the maximum achievable error for each reconstructed region, the total error function is not dependent on the relative sizes of the reconstructed void and fluid regions.

Secondly, the Stream error function differs from the Pilliod error function in the value of the exponent used. While the higher exponent used under the Stream scheme does give slightly better free surface gradient estimations than the Pilliod exponent, in practice the difference in performance between using the two alternative exponents is only slight.

2.4 The Fluid Boundary Flux Calculation

2.4.1 Approximate Integration Method

Ideally, we would like to integrate equation (1) exactly with respect to time to determine donating regions associated with each boundary, and then determine the intersection between these regions and the fluid regions to calculate fluid boundary fluxes. In practice however, such a procedure would be extremely complex because fluid passing though an examined cell boundary during a single time step may have originated from a number of different cells. Consequently, calculating the geometry of the total boundary donating region would be computationally impractical. It is for this reason that an approximate method of integration has been developed to calculate fluid boundary fluxes.

The theory behind the approximate integration method is simple. Each boundary flux is split into a discrete number of streamtubes, each 'tube' representing an equal flux of total fluid and void volume. Fluid streamlines, determined using the velocity field defined by equation (1), define the upper and lower boundaries of each tube. The length of each tube is determined by the length of the computational time step. The number of tubes that each boundary is split into, n_{stream}, is determined by the user—the more tubes the greater the accuracy of integration.

To illustrate this point, Figure 1 shows an example where the velocity at the right boundary of cell i, j is positive, and fluid is fluxing through this

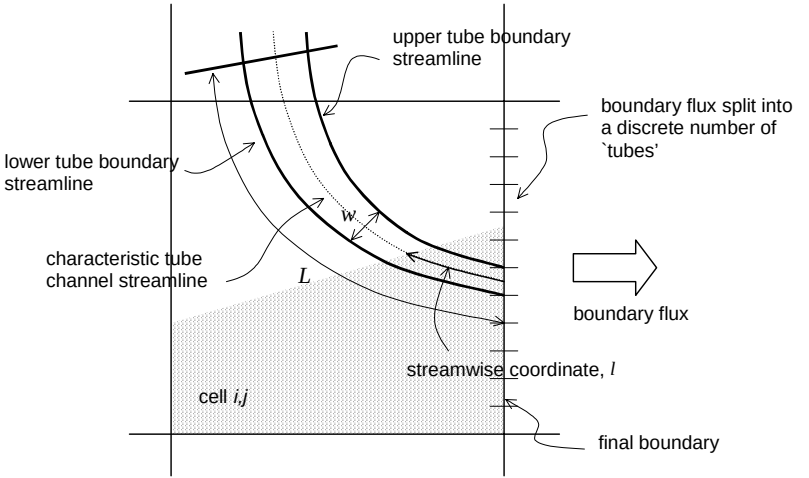


FIGURE 1: An example channel used to calculate the fluid flux over the right boundary of cell i, j.

boundary from cells i, j and $i, j + 1$ during the solution time step. The right boundary of cell i, j is referred to as the 'final' boundary, as it is the last boundary a fluid particle would pass through before entering cell $i + 1, j$.

The volume of fluid in each tube which is fluxed through a particular final

boundary during a time step δt, is approximately

$$V_{\text{fluid in tube}} = \int_0^L w\,(l)\, f\,(l)\, dl, \tag{10}$$

where a cell of unit depth has been assumed. In equation (10), $f\,(l)$ is the VOF function evaluated along the central characteristic streamline of the tube, w is the non-constant width of the tube measured normal to the central tube streamline, l is a streamwise coordinate along the length of the tube directed upstream from the final boundary, and L is the streamwise length of the tube.

To determine the streamwise width of the tube, we note that the total volume of a small section of the tube, located at l and of length Δl, is approximately

$$\Delta V = w\,(l)\,\Delta l. \tag{11}$$

Also, we note that as no fluid may cross the upper and lower boundaries of the tube, the volume contained within this small section is equal to the volume flowrate through the final boundary of the tube multiplied by the time taken for a fluid particle to pass through the small section of the tube. Thus,

$$\Delta V = \frac{u_{i,j} \times \delta y_j}{n_{\text{stream}}} \left[t\,(l) - t\,(l + \Delta l)\right], \tag{12}$$

where t represents the time taken for a fluid particle to flow from the beginning of the tube to position l. Combining equations (11) and (12), rearranging, and taking the limit as $\Delta l \to 0$ yields,

$$w(l) = -\frac{u_{i,j} \times \delta y_j}{n_{\text{stream}}} \lim_{\Delta l \to 0} \frac{1}{\Delta l} \left[t\left(l + \Delta l\right) - t\left(l\right) \right] = -\frac{u_{i,j} \times \delta y_j}{n_{\text{stream}}} \frac{dt}{dl}. \tag{13}$$

Substituting equation (13) into equation (10) gives

$$\begin{aligned} V_{\text{fluid in tube}} &= -\frac{u_{i,j}\delta y_j}{n_{\text{stream}}} \int_0^L f\left(l\right) \frac{dt}{dl} dl = \frac{u_{i,j}\delta y_j}{n_{\text{stream}}} \int_0^{\delta t} f\left(t\right) dt \\ &= \frac{u_{i,j}\delta y_j}{n_{\text{stream}}} t_{\text{total fluid}}, \end{aligned} \tag{14}$$

where $t_{\text{total fluid}}$ is defined as the total time a fluid particle would spend in fluid regions, when moving for time δt along the central tube streamline towards the final cell boundary over a stationary fluid geometry. The algorithm used to calculate $t_{\text{total fluid}}$ is detailed in Harvie and Fletcher [2]. Once individual tube fluxes have been determined, cell averaged VOF values are incremented by summing the fluid fluxes occurring over each of the boundaries of each cell.

2.4.2 Integration Accuracy

An accuracy analysis presented in Harvie and Fletcher [2] shows that the VOF error involved in calculating a fluid flux using the approximate integration

method over a single boundary can be as high as

$$\text{Error}\,(F) = O\left(\frac{u}{2n_{\text{stream}}\delta x}\delta t\right) = O\left(\frac{C}{2n_{\text{stream}}}\right), \tag{15}$$

where the notation $O\,(z)$ specifies 'of the order z', and $C = u\delta t/\delta x$ is the local Courant number. This boundary flux integration error can lead to errors in fluid volume conservation, and in the generation of 'wisps' of fluid in void regions, and 'wisps' of void in fluid regions.

To prevent such problems, two VOF redistribution algorithms are employed under the Stream scheme:

1. A VOF conservation algorithm which corrects F value overshoots and undershoots using a local redistribution algorithm, after all boundary fluxes have been incremented.

2. A 'de-wisping' algorithm which detects the presence of fluid and void 'wisps' after VOF advection has occurred, and locally redistributes fluid to maintain sharp fluid and void interfaces. The VOF fraction which defines both a fluid in void and a void in fluid 'wisp' is determined by the user set variable, F_{wisp}. A level of this variable which is successful in removing all 'wisps' is given by [2]

$$F_{\text{wisp}} = \frac{C}{n_{\text{stream}}}. \tag{16}$$

Provided these algorithms are employed, the Stream scheme conserves fluid volume to the accuracy of the computational floating point arithmetic, and generates no fluid or void 'wisps'.

3 Performance of the Stream Scheme

The performance of the Stream scheme is gauged using a time reversed single vortex advection test, as developed by Rider and Kothe [6]. The velocity field used during this advection test has non-uniform vorticity. Thus, unlike in translation or rotation tests, in the vortex test free surface geometries are sheared and deformed as the fluid moves throughout the computational domain.

In the time reversed single vortex test a cylinder of fluid, of radius 0.15m and centred at (0.5,0.75), is deformed over a period of T s in a velocity field specified by the stream function,

$$\Psi = \frac{1}{\pi} \sin^2(\pi x) \sin^2(\pi y) \cos\left(\frac{\pi t}{T}\right). \tag{17}$$

The direction of the vortex specified by equation (17) reverses over the duration of the test, so that at $t = \frac{T}{2}$ s, the deformation of the cylinder should be at a maximum, and at $t = T$ s, the fluid should have returned to form a cylinder at the initial position. The time reversed vortex test is particularly

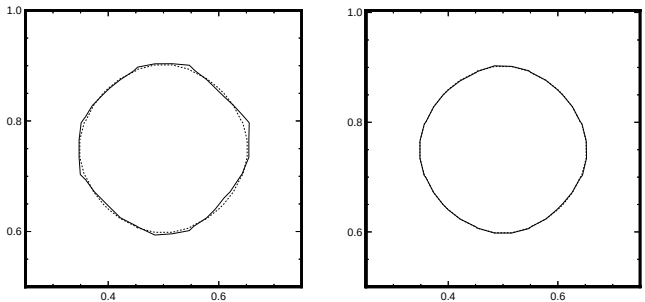
attractive because the accuracy of the advection algorithm can be gauged by simply comparing the initial and final positions of the fluid cylinder.

Figure 2 shows the single vortex test performed using the H-N [3] and Stream VOF advection algorithms over test durations of $T = 0.5$ s and $T = 2.0$ s. As previously discussed, the H-N algorithm is a variation of a piecewise constant advection scheme. Note that all tests presented in this study were performed using a Courant-Friedrichs-Lewy (CFL) number of 1 [6], and used the Stream variables $n_{\text{stream}} = 100$ and $F_{\text{wisp}} = 0.01$.

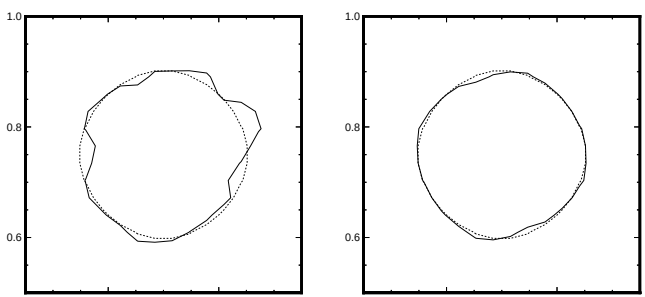
Comparing the different results of Figure 2, it is clear that the Stream scheme performs better than the H-N algorithm in all cases. Also, while not indicated by the contour plots, the H-N algorithm produced a considerable amount of fluid 'wisps' during the two displayed tests. No fluid or void 'wisps' were produced during the Stream algorithm tests.

Figure 3 shows the time reversed single vortex test repeated using the Stream algorithm, but in these cases the duration of the test was extended to $T = 8.0$ s. Cases (A) and (B) show the test performed on a 32×32 mesh, while Cases (C) and (D) show the test performed using a finer 128×128 mesh. Cases (A) and (C) show the fluid geometry at the maximum fluid deformation time of $t = \frac{T}{2}$, while Cases (B) and (D) show the fluid geometry at the conclusion of the test.

As shown in Case (B) of Figure 3, the correlation between the initial and final positions of the cylinder in this long duration test is poor when the 32×32 mesh is employed. Examining Case (A) of the same figure, it



(A) H-N algorithm, T=0.5s (C) Stream algorithm, T=0.5s



(B) H-N algorithm, T=2.0s (D) Stream algorithm, T=2.0s

FIGURE 2: Time reversed single vortex advection tests performed using the H-N and Stream algorithms. All examples are computed on a 32×32 mesh. The dashed contours indicate the initial position, the solid contours the final position.

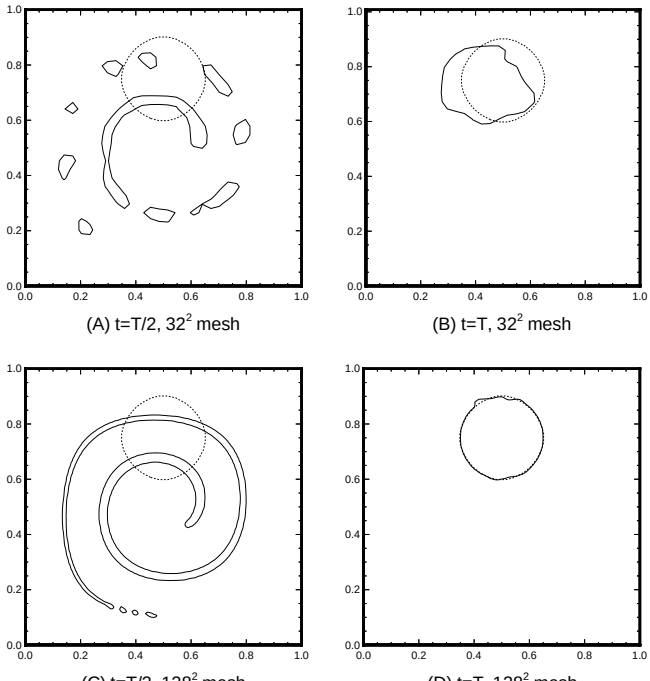


FIGURE 3: Time reversed single vortex advection tests performed by the Stream algorithm over a duration of $T = 8.0$ s. The dashed contours indicate the initial position, the solid contours the final position.

is evident that this poor correlation at later times is due to the breakup of the spiral during the middle stages of the test. Breakup of the spiral occurs because the dimensions of the fluid feature become comparable with the dimensions of the computational cells. Under these circumstances, the free surface interface reconstruction algorithm acts to 'glob' small fluid regions together, causing a type of numerical surface tension.

Cases (C) and (D) of Figure 3 show that when the grid is refined, the Stream scheme is able to accurately calculate the long duration vortex test. Note that the small deformation at the top of the final cylinder, as shown in Case (D), is caused by the minor breakup of the tail of the spiral at intermediate times, as shown in Case (C).

Table 1 shows geometrical test errors calculated using the Stream scheme for a variety of time reversed single vortex tests. The test error is as defined in Rider and Kothe [6] as,

$$E = \sum_{\text{grid}} V_{i,j} \left| F_{i,j}^{\text{initial}} - F_{i,j}^{\text{final}} \right| \tag{18}$$

where V is the volume of each cell. As shown in the table, advection test errors tend to decrease when the cell size is reduced, or if the test duration is reduced. The convergence rate, or spatial order of the scheme, calculated by comparing the errors generated between the same tests performed on differently sized meshes, appears to be approximately two.

A comparison of the accuracy of the Stream scheme against a wider variety of VOF advection algorithms is given in [2].

TABLE 1: Geometrical advection test errors and convergence rates.

Grid	Error $T = 0.5$	Order $T = 0.5$	Error $T = 2.0$	Order $T = 2.0$	Error $T = 8.0$	Order $T = 8.0$
32^2	5.16×10^{-4}		2.34×10^{-3}		4.07×10^{-2}	
		2.31		2.05		2.47
64^2	1.04×10^{-4}		5.64×10^{-4}		7.37×10^{-3}	
		1.68		2.11		2.40
128^2	3.24×10^{-5}		1.30×10^{-4}		1.39×10^{-3}	

4 Conclusions

A new VOF advection algorithm, termed the Stream scheme, has been presented. The algorithm uses a linear piecewise free surface reconstruction method, combined with a unique fully multidimensional boundary flux integration technique. The performance of the new algorithm has been compared against the performance of the H-N algorithm, and it has been found that the Stream scheme is significantly more accurate.

The primary advantages of the Stream scheme over other VOF advection schemes are the accuracy of the advection calculation [2], and the simplicity of the scheme both conceptually, and in application. The simplicity of the algorithm would arguably make the Stream scheme the easiest of the currently available multidimensional advection algorithms to apply to other coordinate systems. An added advantage of the scheme is in the accuracy control afforded to the user. By varying the number of tubes used to perform

the boundary flux integrations, the user can choose between an accurate but computationally expensive simulation, and a less accurate but rapid simulation.

References

[1] D. J. E. Harvie and D. F. Fletcher. A new volume of fluid advection algorithm: The defined donating region scheme. *International Journal for Numerical Methods in Fluids (in press)*, 2000. C693

[2] D. J. E. Harvie and D. F. Fletcher. A new volume of fluid advection algorithm: The stream scheme. *Journal of Computational Physics*, 162:1–32, 2000. C694, C702, C702, C703, C708, C709

[3] 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 Scientific Laboratory, August 1980. C691, C693, C705

[4] W. Noh and P. Woodward. SLIC (simple line interface method). *Lecture Notes in Physics*, 59:330, 1976. C692

[5] J. E. Pilliod Jr. An analysis of piecewise linear interface reconstruction algorithms for volume-of-fluid methods. Master's thesis, University of California at Davis, 1992. C696

[6] W. J. Rider and D. B. Kothe. Reconstructing volume tracking. *Journal of Computational Physics*, 141:112–152, 1998. C692, C692, C693, C693, C704, C705, C708

[7] M. Rudman. Volume-tracking methods for interfacial flow calculations. *International Journal for Numerical Methods in Fluids*, 24:671–691, 1997. C692, C693

[8] D. L. Youngs. Time-dependent multimaterial flow with large fluid distortion. In K. Morton and M. Baines, editors, *Numerical Methods for Fluid Dynamics*, pages 273–285. Academic Press, 1982. C693