INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2001; **35**: 151–172

A new volume of fluid advection algorithm: the defined donating region scheme

Dalton J. E. Harvie[a] and David F. Fletcher[b,*]

[a] *Department of Mechanical and Mechatronic Engineering, University of Sydney, Sydney NSW, Australia*
[b] *Department of Chemical Engineering, University of Sydney, Sydney NSW, Australia*

SUMMARY

This paper presents a new volume of fluid (VOF) advection algorithm, termed the defined donating region (DDR) scheme. The algorithm uses a linear piecewise method of free surface reconstruction, coupled to a fully multi-dimensional method of cell boundary flux integration. The performance of the new scheme has been compared with the performance of a number of alternative schemes using translation, rotation and shear advection tests. The DDR scheme is shown to be generally more accurate than linear constant and flux limited schemes, and comparable with an alternative linear piecewise scheme. The DDR scheme conserves fluid volume rigorously without local redistribution algorithms, and generates no fluid 'flotsam' or other debris, making it ideal in applications where stability of the free surface interface is paramount. Copyright © 2001 John Wiley & Sons, Ltd.

KEY WORDS: advection; free surface stability; volume of fluid

1. INTRODUCTION

The volume of fluid (VOF) method is a convenient and powerful tool for modelling fluid flows that contain a free surface [1]. Under the VOF method, fluid location is recorded using a VOF 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 that contain a free surface.

The VOF method is capable of modelling flows with complex free surface geometries, including flows where fluid volumes separate and combine; yet it is remarkably economical in computational terms, requiring only one mesh sized array for storing the VOF function and an algorithm to advect the function during each computational time step. The method used to advect the VOF function is the subject of this work.

* Correspondence to: Department of Chemical Engineering, University of Sydney, Sydney NSW 2006, Australia.

Received July 1999
Revised January 2000

Excellent reviews of past and present VOF advection methods have been given by Rider and Kothe [2] and Rudman [3], so only a brief overview of some of the methods available will be given here. As a Lagrangian invariant of the fluid, the VOF function, F, satisfies [4]

$$\frac{\partial F}{\partial t} + (V \cdot \nabla)F = 0 \tag{1}$$

Due to the discrete nature of the VOF function, special techniques must be used to difference Equation (1).

One such method is the flux-corrected transport (FCT) algorithm developed by Zalesak [5], which was applied to the process of VOF advection by Rudman [3]. Under the FCT method, Equation (1) is differenced using a combination of diffusive upwind and dispersive downwind first-order difference schemes. The dependence on each scheme is chosen so that the advected solution contains no extrema which were not present in the previous timestep solution. Rudman [3] demonstrated that the FCT–VOF advection method is not as accurate as modern piecewise linear advection methods.

Most VOF advection algorithms are not derived directly from Equation (1) but are based on 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. These types of advection algorithms 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 [2].

VOF advection methods that represent free surface interfaces as lines directed parallel to one of the grid co-ordinates are known as piecewise constant schemes. The simple line interface calculation (SLIC) method of Noh and Woodward [6] and the SURFER method of Lafaurie *et al.* [7] are examples of piecewise constant schemes.

A variation on the piecewise constant theme is the method used in the SOLA–VOF code of Nichols *et al.* [1]. Under the Hirt–Nichols (H–N) scheme, free surface interfaces are orientated in directions parallel to grid co-ordinates, but are also allowed the greater freedom of a stair-shaped profile if local VOF distribution conditions permit. Similar schemes include those developed by Chorin [8] and Barr and Ashurst [9].

The alternative to representing free surface interfaces as lines parallel to one of the grid co-ordinates is to orientate free surface interfaces in a direction perpendicular to the locally evaluated VOF gradient. Thus, free surface interfaces within each cell can acquire any orientation, and the geometrical profile of the fluid can more closely represent the actual fluid geometry. Such schemes are known as piecewise linear schemes, and include those developed by Rider and Kothe [2], Debar [10], Youngs [11], Ashgriz and Poo [12], Puckett *et al.* [13], and Harvie and Fletcher [17]. These schemes tend to be more complex than their piecewise constant cousins, but have been shown to be significantly more accurate [2,3].

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 co-ordinate direction, often with some type of limiter employed to reduce possible undershoots or overshoots occurring in cell VOF values. In some operator split schemes, free surfaces are reconstructed between integrations in each of the co-ordinate directions. The early Youngs [11] algorithm is an example of an operator split scheme.

Multi-dimensional schemes can be more efficient in calculating cell boundary fluxes than operator split schemes [2]. Under multi-dimensional schemes, cell boundary fluxes are calculated with a dependence between fluxes calculated in each of the co-ordinate directions. Only one free surface reconstruction per time step is required by multi-dimensional schemes. Example multi-dimensional schemes include those developed by Rider and Kothe [2], Puckett *et al.* [13], and Harvie and Fletcher [17].

The defined donating region (DDR) scheme, as developed in this study, is a piecewise linear scheme with cell boundary fluxes integrated using a new multi-dimensional method. While the scheme shares similarities with existing multi-dimensional schemes, important differences in the donating region definition used by the DDR scheme ensures that it is unique, possessing its own individual strengths and weaknesses.

In this paper, we detail the new VOF advection scheme. This is accomplished in three sections. In the first, the donating regions within each cell are defined; in the second, the free surface reconstruction method is outlined; and in the last, the multi-dimensional integration technique is detailed. A comparison of the performance of the DDR algorithm against several other VOF advection schemes is then given, using for comparison several simple VOF advection tests.

2. THE DEFINED DONATING REGION ALGORITHM

2.1. Defined donating regions

Under the DDR method, each boundary has associated with it a defined region from which fluid can cross the boundary, or be donated. The actual fluid flux over the boundary is then the intersection of the fluid region contained in the cell with the defined donating region. To conserve fluid volume and to preserve fluid geometries, such a scheme must satisfy two requirements

1. donating regions for any two boundaries must not overlap, otherwise VOF conservation is not assured; and
2. donating regions must contain a total volume equal to the volume of fluid and void, which is fluxed over the associated boundary during the computational time step.

The method used to define a donating region is illustrated in Figure 1. In this Cartesian example of unit depth, fluid leaves the cell over the right boundary with velocity u_R, enters the cell over the bottom boundary with velocity v_B and leaves the cell over the top boundary with velocity v_T. Thus, for this example cell, the right and top boundaries are donating boundaries, which require defined donating regions, while the bottom boundary is an accepting boundary requiring no defined region within the cell. There is no fluid flow over the left boundary. As

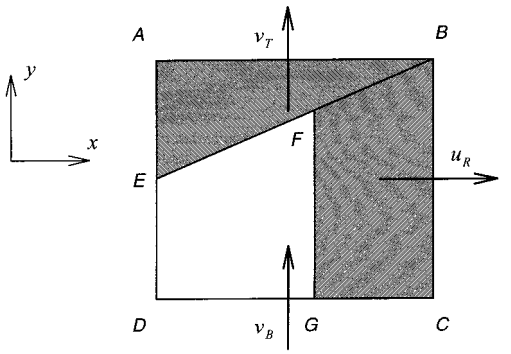

Figure 1. Variables used to define the donating region within an example cell.

shown in Figure 1, each donating region is defined as a trapezium having two faces parallel, or in the limiting case of a trapezium with one zero length parallel side, a triangle.

The velocity of each adjacent boundary is used to determine the gradient of each of the non-parallel trapezium sides. Considering the donating region *FBCG* attached to the right donating boundary *BC* in Figure 1, we note that the adjacent lower boundary *DC* is an accepting boundary, and thus requires no donating region in this cell. Consequently, the lower trapezium boundary *GC* is parallel to the lower boundary *DC*. Conversely, the upper donating boundary *AB* does require an attached donating region, so the gradient of this trapezium side *FB* is set to the inverse ratio of velocities of the two adjacent boundaries. In this case

$$\left.\frac{\mathrm{d}y}{\mathrm{d}x}\right|_{FB} = \frac{v_\mathrm{T}}{u_\mathrm{R}} \tag{2}$$

The idea behind this gradient is as follows. Given a uniform flow field throughout the cell having horizontal and vertical components of u_R and v_T respectively, fluid residing on the line *EB* would pass through the point *B* when exiting the cell. Fluid above *EB* would pass through the top boundary *AB*, while fluid below *EB* would pass through the right boundary *BC*. Thus, the line *EB* defines the intersection of the two donating regions.

Once the gradients of the two non-parallel sides have been set, the position of the internal parallel side is set to give the donating region volume equal to the total fluid and void flux through the donating boundary. To locate the trapezium side *FG* in Figure 1, the volume V_{FBCG} of the trapezium *FBGC* must satisfy

$$V_{FBCG} = \frac{1}{2}(Y_{BC} + Y_{FG})X_{GC} = u_\mathrm{R}\,\delta y\,\delta t \tag{3}$$

where δy is the height of the cell, δt is the time step and X and Y are lengths in the horizontal and vertical directions respectively.

Further examples of donating regions for different combinations of fluid boundary velocities are given in Figure 2. Case (A) shows a square cell with equal magnitude velocities over each boundary. The contents of the entire cell is removed within the time step, the lower right half out the right boundary, the upper left half out the top boundary. Case (B) shows two adjacent donating boundaries with unequal magnitudes. Three donating boundaries are shown in case (C). Case (D) shows two opposite donating boundaries of equal magnitude which, as in case (A), remove the entire contents of the cell within the time step. Case (E) demonstrates the donating regions in a cell with unequal cell dimensions.

2.2. Stability analysis

A stability analysis of the donating region model is undertaken to provide the maximum stable time step. The analysis also shows that for any given set of cell boundary velocities satisfying the discretized continuity equations, and for any rectangular cell dimensions, donating regions within the cell can be defined.

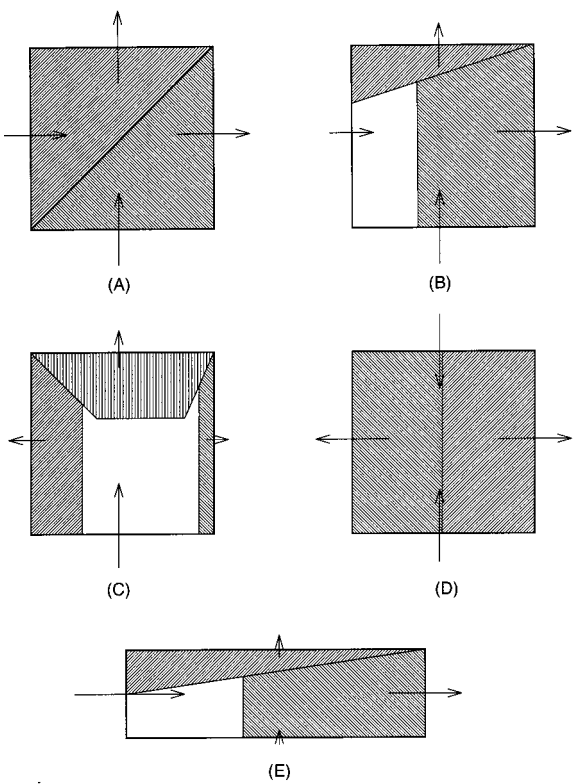

Figure 2. Example donating regions defined using boundary velocities.

For stability, the volume of fluid and void entering the cell during a time step must be less than the volume of the cell, otherwise any geometrical information concerning the orientation and volume of fluid within the cell becomes meaningless. Representing the total volume of fluid and void flowing into the cell as V_{in}, the total volume flowing out as V_{out} and the total volume of the cell as V_{cell}, we have

$$V_{\text{in}} \leq V_{\text{cell}} \tag{4}$$

For conservation of fluid volume, $V_{\text{in}} = V_{\text{out}}$, and so

$$V_{\text{out}} \leq V_{\text{cell}} \tag{5}$$

Combining Equations (4) and (5) gives

$$V_{\text{in}} + V_{\text{out}} \leq 2V_{\text{cell}} \tag{6}$$

Noting that the sum of volume inflow and volume outflow is the total volume flow over all cell boundaries, and using the velocity notation defined in Figure 3 for the illustrated Cartesian cell

$$|V_{\text{T}}| + |V_{\text{R}}| + |V_{\text{B}}| + |V_{\text{L}}| \leq 2V_{\text{cell}} \tag{7}$$

$$\therefore\ |v_{\text{T}}|\frac{\delta t}{\delta y} + |u_{\text{R}}|\frac{\delta t}{\delta x} + |v_{\text{B}}|\frac{\delta t}{\delta y} + |u_{\text{L}}|\frac{\delta t}{\delta x} \leq 2 \tag{8}$$

Equation (8) is satisfied at each boundary if the time step satisfies either

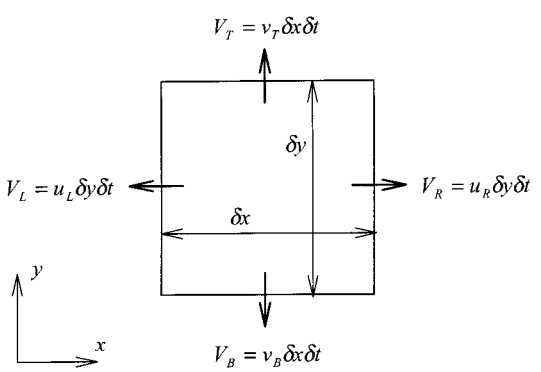

Figure 3. Variables used in the stability analysis.

$$|u_{i+1/2}|\delta t \leq \frac{1}{2}\min(\delta x_i, \delta x_{i+1}) \tag{9}$$

for the x-direction, or equivalently for the y-direction

$$|v_{j+1/2}|\delta t \leq \frac{1}{2}\min(\delta y_j, \delta y_{j+1}) \tag{10}$$

Integer subscripts in these equations imply cell-centred quantities. Subscripts containing an integer plus or minus a half refer to cell boundary quantities located above or below the indicated cell respectively.

Equations (9) and (10) are a form of the Courant condition. In cylindrical co-ordinates, Equation (9) becomes

$$|u_{i+1/2}|\delta t \leq \frac{1}{4x_{i+1/2}}\min(x^2_{i+1/2} - x^2_{i-1/2}, x^2_{i+3/2} - x^2_{i+1/2}) \tag{11}$$

where x is now the radial co-ordinate, and Equation (10) remains unchanged.

As previously discussed, donating regions must not overlap, and must contain the total VOF and void that is fluxed over the associated boundary during the time step. To demonstrate that donating regions can always be defined within a cell, we examine the four possible cases of donating boundary layout.

2.2.1. One donating boundary. Equations (9)–(11) are equivalent to

$$|V_{\text{boundary}}| \leq \frac{1}{2} V_{\text{cell}} \tag{12}$$

where V_{boundary} is the total volume flux over any cell boundary. For the case of only one donating boundary in the cell, as shown in Figure 4(A), Equation (12) shows that the maximum volume the donating region can have is half the cell volume. Thus, in this trivial one-dimensional case, the rectangular region *EBCF* can always be defined.

2.2.2. Two adjacent donating boundaries. We wish to show that given the region boundary gradient assumption equation (2) and stability criterion equation (12), the donating regions can always be defined. Examining the example shown in Figure 4(B), where $v_T/u_R \leq \delta y/\delta x$, the volume of the triangular region is

$$V_{ABE} = \frac{1}{2} Y_{AE} X_{AB} = \frac{1}{2} \delta x Y_{AE} \tag{13}$$

Employing the gradient assumption equation (2)

$$\left.\frac{\mathrm{d}y}{\mathrm{d}x}\right|_{FB} = \frac{Y_{AE}}{X_{AB}} = \frac{Y_{AE}}{\delta x} = \frac{v_T}{u_R} = \frac{V_T}{\delta x\, \delta t} \cdot \frac{\delta y\, \delta t}{V_R} \tag{14}$$

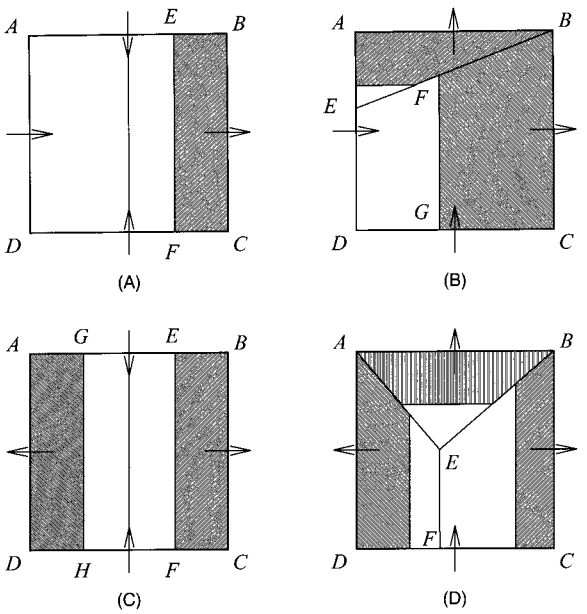

Figure 4. Four possible donating boundary layouts.

$$\therefore \frac{V_{\mathrm{T}}}{V_{\mathrm{R}}} = \frac{Y_{AE}}{\delta y} \tag{15}$$

Substitution into Equation (13) yields

$$V_{ABE} = \frac{1}{2}\,\delta x\,\delta y\,\frac{V_{\mathrm{T}}}{V_{\mathrm{R}}} = \frac{1}{2}\,V_{\mathrm{cell}}\,\frac{V_{\mathrm{T}}}{V_{\mathrm{R}}} \tag{16}$$

Employing the stability criterion equation (12) for the right boundary

$$V_{\mathrm{R}} \leq \frac{1}{2}\,V_{\mathrm{cell}} \tag{17}$$

gives

$$V_{\mathrm{T}} \leq V_{ABE} \tag{18}$$

Further, if the stability assumption is also applied at the top boundary

$$V_{\mathrm{T}} \leq \frac{1}{2} V_{\mathrm{cell}} \tag{19}$$

noting that the volume of the trapezium is

$$V_{EBCD} = \frac{1}{2}(Y_{ED} + Y_{BC}) X_{AB} = \frac{1}{2}(2\delta y - Y_{AE})\delta x \tag{20}$$

the right volume flux is

$$V_{\mathrm{R}} \leq V_{EBCD}\left(\frac{\delta y^2}{(2\delta y - Y_{AE}) Y_{AE}}\right) \tag{21}$$

The bracketed term of Equation (21) has a maximum value of 1 for real values of Y_{AE}, giving

$$V_{\mathrm{R}} \leq V_{EBCD} \tag{22}$$

From Equations (18) and (22), the VOF fluxed through each boundary is contained within each associated region. Thus, provided the fluxes satisfy the stability criterion equation (12), both donating regions are defined. The choice of boundaries for this example was arbitrary, so the proof holds for any two adjacent donating boundaries.

2.2.3. Two opposite donating boundaries. This case is an extension of the trivial one-dimensional, one donating boundary case. As shown in Figure 4(C), provided the stability criterion equation (12) is met, the rectangular regions $AGHD$ and $EDCF$ can be defined.

2.2.4. Three adjacent donating boundaries. The maximum volumes for the three donating regions illustrated in Figure 4(D) are

$$V_{ABE} = \frac{1}{2}\delta x(\delta y - Y_{EF}) \tag{23}$$

$$V_{AEFD} = \frac{1}{2} X_{DF}(\delta y + Y_{EF}) \tag{24}$$

and

$$V_{EBCF} = \frac{1}{2}(\delta x - X_{DF})(\delta y + Y_{EF}) \tag{25}$$

For the donating regions to be defined, their volumes must be contained within these maxima.

Employing the region boundary gradient assumption equation (2) at EB and AE gives

$$V_{\mathrm{T}} = V_{\mathrm{R}} \frac{\delta x(\delta y - Y_{EF})}{\delta y(\delta x - X_{DF})} \tag{26}$$

and

$$V_{\mathrm{L}} = V_{\mathrm{R}} \frac{\delta x(\delta y - Y_{EF})}{\delta y X_{DF}} \tag{27}$$

respectively. Combining Equations (26) and (27) for the left volume flux gives

$$V_{\mathrm{L}} = V_{\mathrm{R}} \frac{X_{DF}}{\delta x - X_{DF}} \tag{28}$$

Now, examining the stability criterion equation (12) for the lower boundary

$$|V_{\mathrm{B}}| \leq \frac{1}{2} V_{\mathrm{cell}} \tag{29}$$

The lower boundary is the only boundary with fluid entering the cell, so noting that $V_{\mathrm{in}} = V_{\mathrm{out}}$, we have

$$V_{\mathrm{L}} + V_{\mathrm{T}} + V_{\mathrm{R}} \leq \frac{1}{2} V_{\mathrm{cell}} \tag{30}$$

Combining Equations (25), (26), (28) and (30) gives for the volume flux over the right boundary

$$V_{\mathrm{R}} \leq V_{EBCF} \left[2 - \frac{Y_{EF}}{\delta y} + \left(\frac{Y_{EF}}{\delta y} \right)^2 \right]^{-1} \tag{31}$$

Examination of the bracketed denominator on the right shows that it has a minimum of 1.75 for real values of Y_{EF}. Thus

$$V_{\mathrm{R}} \leq V_{EBCF} \tag{32}$$

Turning to the left region, we have from Equations (24), (28) and (31)

$$V_{\mathrm{L}} \leq V_{AEFD} \left[2 - \frac{Y_{EF}}{\delta y} + \left(\frac{Y_{EF}}{\delta y} \right)^2 \right]^{-1} \tag{33}$$

$$\therefore\ V_{\mathrm{L}} \leq V_{AEFD} \tag{34}$$

For the top region, we have from Equations (23), (26) and (31)

$$V_{\mathrm{T}} \leq V_{AED}\left[1+\frac{1-\left(\frac{Y_{EF}}{\delta y}\right)^2}{1-\frac{Y_{EF}}{\delta y}}\right]^{-1} \tag{35}$$

The bracketed term has a minimum of 2 for real values of Y_{EF}, so

$$V_{\mathrm{T}} \leq V_{AED} \tag{36}$$

Equations (32), (34) and (36) show that donating regions can always be defined within a three donating boundary cell, provided the stability criterion equation (12) is met. As a cell cannot possess four donating boundaries, the existence of donating regions for any combination of donating boundaries has been proven.

In cylindrical co-ordinates, the total boundary fluxes are given by

$$V_{\mathrm{T}} = \pi(x_i^2 - x_{i-1}^2)v_{\mathrm{T}} \tag{37}$$

$$V_{\mathrm{B}} = \pi(x_i^2 - x_{i-1}^2)v_{\mathrm{B}} \tag{38}$$

$$V_{\mathrm{R}} = 2\pi x_i \delta y u_{\mathrm{R}} \tag{39}$$

$$V_{\mathrm{L}} = 2\pi x_{i-1} \delta y u_{\mathrm{L}} \tag{40}$$

and the existence of donating regions proof still applies. A small error in calculating the donating region geometry and fluid position within cylindrical cells is introduced by assuming the internal geometry of the cell is Cartesian; however, this error decreases with cell dimensions and is negligible in all practical cases. This error does not affect VOF conservation.

2.3. Free surface reconstruction

As a piecewise linear scheme, the free surface reconstruction method used by the DDR scheme is fairly standard. Free surface orientations are evaluated in each cell by calculating the local gradient of the VOF function. In the present study, two interface gradient methods are employed.

1. Youngs method [14]. Under this method the VOF gradient is calculated using a simple difference expression evaluated over a 3×3 cell kernel. Youngs method has been shown to produce approximately first-order surface reconstructions [2]. The implementation used here is taken from Reference [4].
2. Puckett method [15]. Under the Puckett method, each interface orientation within each cell has associated with it an error function. When this error function is minimized, we find the optimal free surface orientation. The implementation of the Puckett method used here is taken from Reference [17]. The Puckett method is more computationally expensive than the Youngs method; however, it has been shown to produce higher-order accuracy free surface reconstructions [2].

The performances of the two gradient calculation methods are compared in the next section.

Once the gradients of the interfaces within each cell have been calculated, the position of the interfaces are set to equate the volume of fluid beneath each interface to the volume of fluid contained in each cell. Details of the algorithm employed to accomplish this can be found in Reference [16].

Note that in cylindrical co-ordinates the interface positioning is accomplished using internal cell volumes calculated in Cartesian co-ordinates. This introduces a small interface gradient error, but like the cylindrical treatment of the defined donating regions, the error is negligible for all practical cases and does not affect fluid conservation.

2.4. Integration of the VOF fluxes

The amount of flux donated over each boundary is calculated as the volume intersection between the defined donating region of the donating cell and the reconstructed position of fluid within that cell.

In the example of Figure 5, fluid lies within the quadrilateral *EGDH*. The velocity over boundary *AD* is positive, so the boundary has a donating region associated with it, namely the trapezium *ADCB*. The VOF fluxed over *AD* is the intersection between the fluid region *EGDH* and the donating region *ADCB*, the region *FGDC*. In this example, the amount fluxed is

$$\delta F_{\mathrm{D}} = \frac{1}{2\delta x\, \delta y}\,(Y_{FC} + Y_{GD})X_{CD} \tag{41}$$

where δx and δy are the cell dimensions.

In Cartesian co-ordinates, the change in the VOF value of the donating cell is calculated using

$$F^{*}_{\mathrm{D}} = F_{\mathrm{D}} - \delta F_{\mathrm{D}} \tag{42}$$

while the VOF value for the accepting cell is incremented using either

$$F^{*}_{\mathrm{A}} = F_{\mathrm{A}} + \frac{\delta F_{\mathrm{D}}\, \delta x_{\mathrm{D}}}{\delta x_{\mathrm{A}}} \tag{43}$$

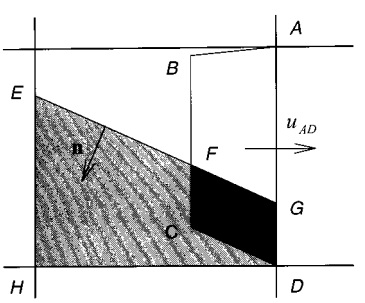

Figure 5. An example flux calculation over a right donating boundary.

in the x-direction, or in the y-direction

$$F_A^* = F_A + \frac{\delta F_D \, \delta y_D}{\delta y_A} \tag{44}$$

Here F^* are the incremented VOF values, δF_D is the calculated VOF loss from the donating cell, δx and δy are cell dimensions and the subscripts A and D refer to accepting and donating cells respectively.

In cylindrical co-ordinates, Equation (44) still holds in the y-direction, but in the radial direction, Equation (43) is replaced by

$$F_A^* = F_A + \delta F_D \frac{(x_{D+}^2 - x_{D-}^2)}{(x_{A+}^2 - x_{A-}^2)} \tag{45}$$

where the previous notation is supplemented by x_+ and x_-, the positions of the boundaries on either side of the relevant cell. Equation (42) still holds in both directions.

3. PERFORMANCE OF THE DEFINED DONATING REGION SCHEME

In this section the performance of the DDR scheme is compared against other VOF advection schemes using a variety of advection tests.

3.1. Translation test—the box

The simplest advection algorithm test involves translating a geometric shape around the computational domain. Under such a test, the geometric shape should remain intact, and the total amount of fluid within the region should be conserved. The test examined here, that of a box being translated by a uniform and constant velocity field, was chosen to highlight some of the problems existing with the original H–N algorithm.

The two-dimensional Cartesian regions shown in Figure 6 have dimensions of 1×1 m^2 and are composed of 10000 equally sized cells, each 0.01×0.01 m^2. A square block of fluid, of dimensions 0.1×0.1 m^2, moves with equal horizontal and vertical velocities of 1 m s^{-1} towards the top right-hand corner of the computational domain. Figure 6 shows the fluid position computed using the H–N and DDR algorithms every 0.1 s until 0.7 s. The exact solution is also shown. The computational time step used in these calculations was 1×10^{-3} s, yielding a Courant number of 0.1.

As shown in Figure 6, the VOF contours calculated by the H–N algorithm show significant diffusion of fluid in a direction normal to the velocity of the fluid. While this diffusion tends to decrease with decreasing time step, the time step used in the example, 1×10^{-3} s, is significantly smaller than the maximum time step for stability of 5×10^{-3}s [1].

In comparison, the VOF contours calculated using the DDR method in Figure 6 show that the DDR algorithm is more accurate in predicting fluid translation. The shape of the box at the end of the DDR calculations has not 'flattened' tangentially to the extent that the H–N

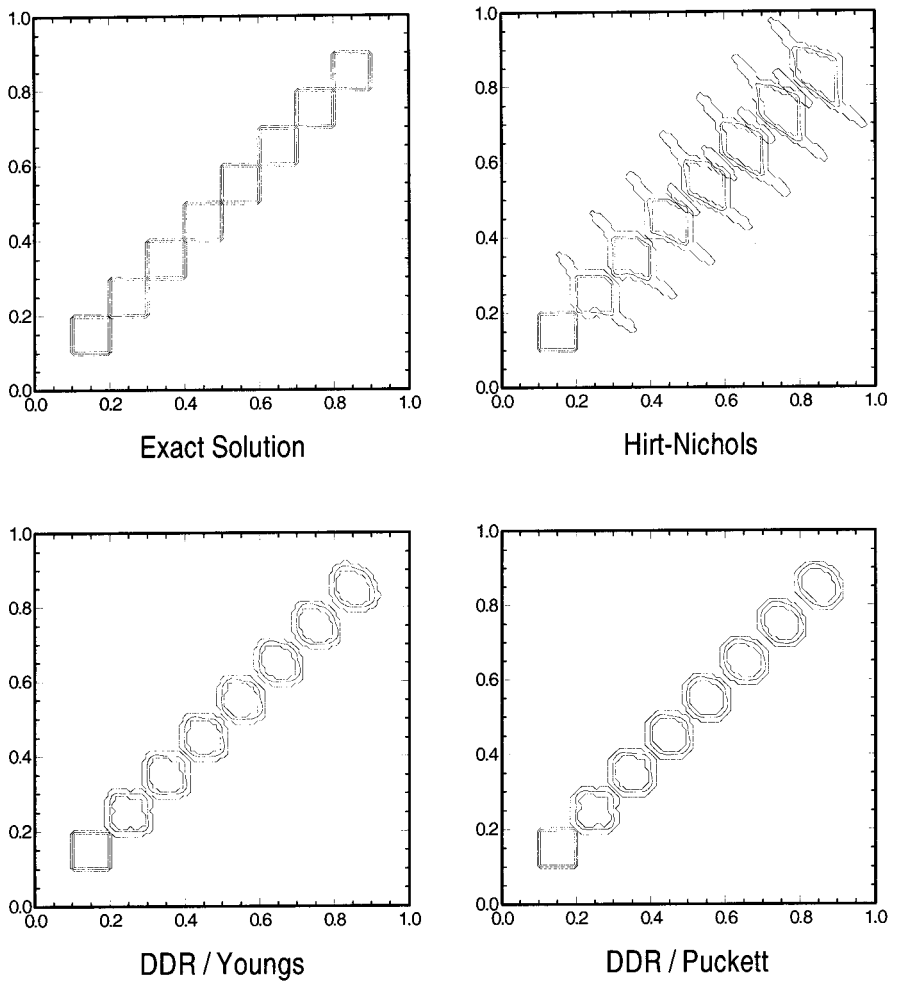

Figure 6. The translation test box problem results calculated using the H–N algorithm, the DDR algorithm using the Youngs interface gradient method and the DDR algorithm using the Puckett interface gradient method. The exact solution is also shown. In each case three VOF contours are shown; ε_F, 0.5 and $1-\varepsilon_F$, where $\varepsilon_F = 1 \times 10^{-6}$.

test box did, and the total distance moved by the centre of the box is predicted more accurately. Also, there is no wisp generation evident at the new box corners moving parallel to the fluid velocity, despite the DDR algorithm not including any wisp suppressing features. It is not clear from the figures which method of gradient interface calculation produces the more accurate results.

Despite the improvement, some diffusion of the box shape has still occurred under the DDR scheme. This is evidenced by rounding of the box corners and some spreading of the box in a direction normal to the velocity direction. The rounding of the corners is a feature of the finite grid size and interface reconstruction method used in calculating boundary fluid fluxes—such rounding is reduced as the mesh size is refined. The minor spreading of the box shape is caused by the staggered location of the fluid velocity components. This spreading is reduced with a decrease in the computational time step size, and also with grid refinement.

Table I shows error functions generated during each translation test. The error function is here defined as

$$E = \frac{\sum_{i,j} |F^n_{i,j} - F^e_{i,j}|}{\sum_{i,j} F^e_{i,j}} \tag{46}$$

where F^n and F^e are the calculated and exact VOF functions at the completion of the test respectively. As shown, the error function generated by the H–N algorithm is substantially larger than the error functions generated by the DDR algorithm. The Youngs method of interface gradient calculation appears to generate a lower error function than the Puckett method of interface gradient calculation; however, the difference between the two results is only slight.

Figure 7 shows the total area of fluid within the two-dimensional computational region, as calculated by the H–N and DDR algorithms, as a function of time. Concerningly, under the H–N algorithm, this fluid area increases to approximately 130 per cent of the initial amount by the completion of the test. This shows that the total amount of fluid is not being conserved by the H–N algorithm, and that Equation (1) is not being satisfied. In comparison, the DDR scheme conserves fluid area to within the precision of the VOF surface cell indicator, $\varepsilon_F = 1 \times 10^{-6}$.

Finally, Table II shows the computational times required to perform the translation tests. All figures have been normalized against the time required by the H–N algorithm. As shown, the H–N algorithm and DDR algorithm employing the Youngs interface gradient method required a similar amount of time to complete the test. The DDR algorithm employing the Puckett interface gradient method took a slightly longer period of time. The Puckett interface method is more computationally expensive than the Youngs method, as the Puckett method is iterative, requiring nine cell interface reconstructions in each cell for each interface orientation iteration. Note that in actual fluid calculations, however, far greater computational time is generally spent inverting the pressure field matrix than advecting the VOF function, so the relative expense of each VOF advection technique is of only minor importance.

Table I. Error functions generated during the box translation tests.

H–N	DDR/Youngs	DDR/Puckett
0.539	0.190	0.216

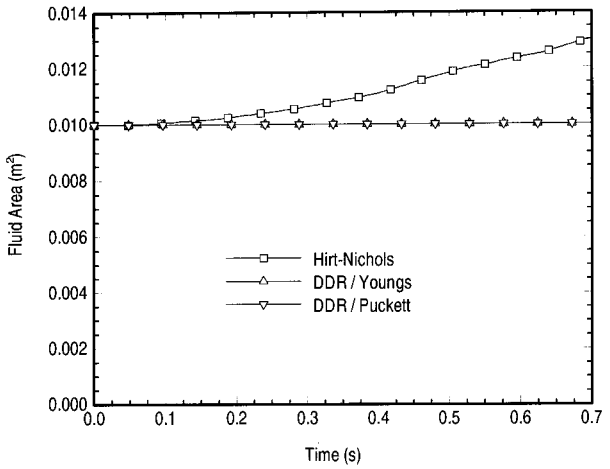

Figure 7. The total fluid areas calculated by the H–N and DDR algorithms during the box translation test.

Table II. Non-dimensionalized CPU time consumed during the box translation tests.

H–N	DDR/Youngs	DDR/Puckett
1.0	1.0	1.3

3.2. Rotation test—the Rudman–Zalesak slotted disk

The Zalesak slotted disk test has become a benchmark test for comparison of scalar advection algorithms. The test involves rotating a slotted disk through one complete revolution within the computational domain under the action of a uniform vorticity velocity field. Advection algorithm accuracy can be gauged by comparing the initial and final positions of the disk.

To allow comparison of the DDR algorithm against other VOF advection algorithms, the form of the Zalesak test performed here is taken from Reference [3]. A computational domain of dimensions 4×4 m^2 is composed of 200×200 uniformly sized square cells. The disk has a diameter of 1 m, and one revolution of the disk is completed in exactly 2524 time steps. This time step corresponds to a Courant number, based on the maximum co-ordinate velocity within the domain, of approximately 0.25. Further details of the form of the test can be found in Reference [3].

Figure 8 shows graphical results of the Rudman–Zalesak test performed using the DDR algorithm. These figures may be compared with figures published in Reference [3], where the same test was performed using the H–N [1], FCT–VOF [3], SLIC [6], and Youngs [11] algorithms. Note that an operator split implementation of the H–N algorithm was used by Rudman [3] in these tests rather than the original multi-dimensional implementation as used in the previous section of the present study.

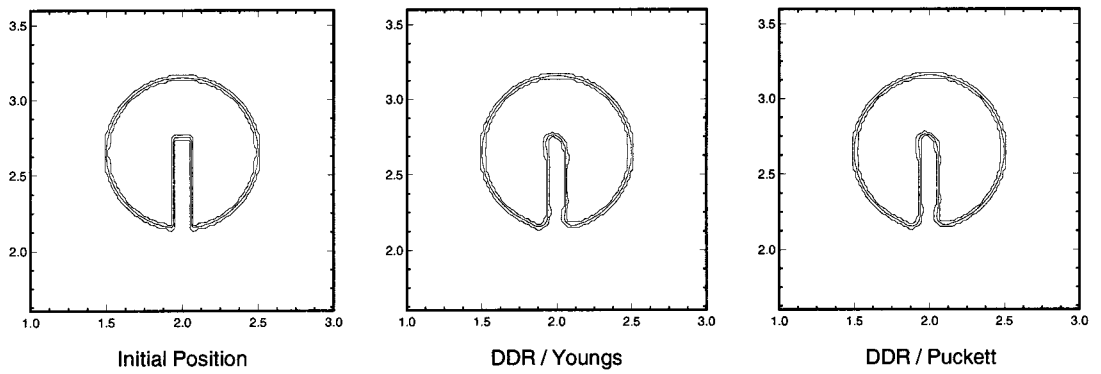

Figure 8. The Rudman–Zalesak test performed using the DDR algorithm. The initial fluid position, and results produced using the Youngs and Puckett interface gradient calculation methods, are displayed. Consistent with the figures shown in Reference [3], in each case three VOF contours are shown; 0.025, 0.5 and 0.975.

Comparing Figure 8 against the figures published in Reference [3] shows that the DDR algorithm provides a similar level of accuracy to the Youngs algorithm, and a superior level of accuracy to the piecewise constant and flux limited schemes. There is little visible difference between the DDR results produced using the two different free surface interface calculation methods.

Table III shows error functions calculated during the Rudman–Zalesak test using a variety of VOF advection methods. All results, expect for those concerning the DDR algorithm, are taken from Reference [3], and the error function is again defined by Equation (46).

As shown in Table III, the Youngs algorithm, which uses the Youngs method of free surface gradient calculation, produces the best results in this test. The DDR algorithm, also employing the Youngs method of free surface gradient calculation, is less accurate, but more accurate than the piecewise constant or flux limited algorithms. The DDR algorithm employing the Puckett method of free surface gradient calculation is slightly more accurate than the same algorithm employing the Youngs method of free surface gradient calculation, but still less accurate than the original Youngs algorithm.

Table III. Error functions generated during the Rudman–Zalesak slotted disk tests.

SLIC	H–N	FCT–VOF	Youngs	DDR/Youngs	DDR/Puckett
8.38×10^{-2}	9.62×10^{-2}	3.29×10^{-2}	1.09×10^{-2}	1.56×10^{-2}	1.50×10^{-2}

All results shown, except those involving the DDR algorithm, are taken from Reference [3].

3.3. Shear test—the Rudman vortex

The final advection test examined in this study employs a non-uniform vorticity velocity field, which stretches and shears free surface interfaces as fluid is translated throughout the computational domain. To facilitate comparison of the DDR algorithm against other VOF advection algorithms, the form of this test is again taken from Reference [3].

A two-dimensional computational domain of dimensions $\pi \times \pi$ m^2, and composed of 100×100 uniformly sized square cells, is used in the shear test. The velocity field is specified by

$$u = A \sin x \cos y \quad \text{and} \quad v = -A \cos x \sin y \tag{47}$$

where A equals 1 for the first N computational time steps, and -1 for the second N time steps. The initial fluid geometry is a circle of radius $\pi/5$, and a Courant number of 0.25, based on the maximum co-ordinate velocity within the computational domain, is used.

The velocity field specified by Equation (47) is time reversed, so that after $2N$ computational time steps an exact advection algorithm would return the fluid to the starting location. Thus, advection algorithm accuracy in the shear test can again be gauged by comparing the initial and final positions of the fluid.

Figure 9 shows graphical results for the Rudman shear test performed using the DDR algorithm, and using four different test durations. Comparing these figures to those published in Reference [3], the DDR algorithm appears to be more accurate than the piecewise linear or flux limited algorithms, but less accurate than the original Youngs algorithm. The errors generated during these tests, which are shown in Table IV and again defined by Equation (46), generally support these observations.

It is interesting that during the longer duration tests shown in Figure 9, the 'tail' of the fluid spiral tends to 'breakup' at intermediate test times. Breakup of the spiral occurs because the width of the fluid form becomes comparable with the computational cell dimension. In such cases, the interface reconstruction technique tends to arrange the small amounts of fluid in each cell as close together as possible, causing the fluid to 'glob'. This process can be thought of as numerical surface tension and has been previously observed by other researches [2,17]. As shown, its effect is most pronounced when the dimensions of the fluid region are similar to or smaller than the dimensions of the computational cells.

Comparing the shear test results generated using the Youngs and Puckett interface gradient schemes, the least amount of spiral breakup occurs when using the more complex Puckett interface gradient scheme. As shown in Table IV, this is reflected in lower shear test error functions generated under the Puckett interface gradient scheme than under Youngs interface scheme.

In order to examine the spatial convergence rate of the DDR algorithm, the 10π s duration shear test was repeated using four different mesh sizes. A Courant number of 0.25 was used in all of the tests, so that the computational time step was reduced as the mesh was refined. The error functions generated during these tests, and the associated convergence rates, are shown in Table V. The results indicate that the average spatial convergence rate of the DDR algorithm over all of the tests was approximately 1.4. Also, the convergence rate is dependent

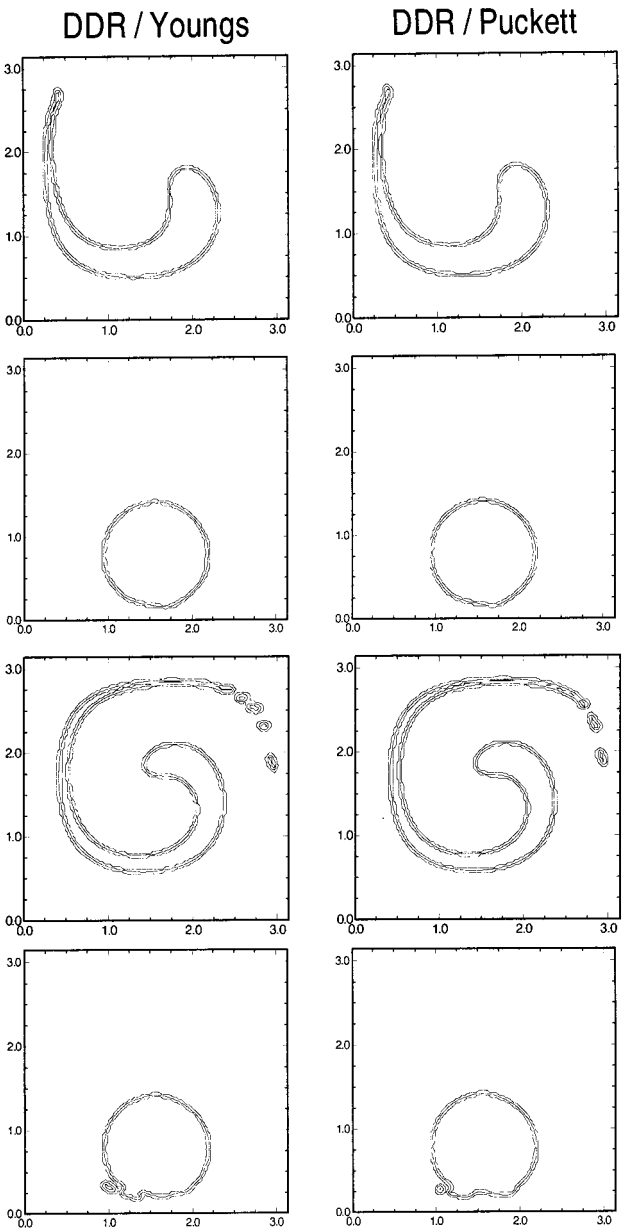

Figure 9. The Rudman shear test performed using the DDR algorithm. Results produced using the Youngs and Puckett interface gradient calculation methods are displayed. Consistent with the figures shown in Reference [3], in each case three VOF contours are shown; 0.025, 0.5 and 0.975.

Table IV. Error functions generated during the Rudman shear tests.

N	SLIC	H–N	FCT–VOF	Youngs	DDR/Youngs	DDR/Puckett
250	2.27×10^{-2}	3.24×10^{-2}	1.94×10^{-2}	2.61×10^{-3}	4.44×10^{-3}	3.16×10^{-3}
500	3.30×10^{-2}	4.00×10^{-2}	2.35×10^{-2}	5.12×10^{-3}	7.35×10^{-3}	7.13×10^{-3}
1000	4.59×10^{-2}	6.66×10^{-2}	3.14×10^{-2}	8.60×10^{-3}	1.25×10^{-2}	1.19×10^{-2}
2000	9.02×10^{-2}	1.09×10^{-1}	1.44×10^{-1}	3.85×10^{-2}	5.15×10^{-2}	4.51×10^{-2}

All results shown, except those involving the DDR algorithm, are taken from Reference [3].

on the method of interface gradient calculation, with the Puckett method displaying a slightly higher rate than the Youngs method.

Examining the results of the three advection tests that have been performed in this study, it appears that the accuracy of the DDR algorithm is significantly greater than that of the piecewise constant and flux limited schemes, but slightly lower than that of the original Youngs algorithm. However, the DDR algorithm does have two advantages over the Youngs algorithm, and other comparable piecewise linear algorithms, which justifies its use in modern multi-phase flow applications.

1. The DDR scheme rigorously conserves fluid volume, despite it containing no VOF 'overshoot' or 'undershoot' correction algorithms. This feature is a result of the defined donating regions possessing no overlapping areas.
2. The DDR scheme produces no fluid 'flotsam', in spite of containing no algorithms used to suppress such debris. This feature is a result of the defined donating region geometry, which tends to fill transition region cells before depositing fluid into empty cells, and similarly tends to empty transition region cells before removing fluid from full cells.

Table V. Error functions and convergence rates for the 10π s duration Rudman shear test, shown as a function of grid refinement.

Grid	DDR/Youngs		DDR/Puckett	
	Error	Order	Error	Order
50^2	1.36×10^{-1}		1.48×10^{-1}	
		1.40		1.71
100^2	5.15×10^{-2}		4.51×10^{-2}	
		1.21		1.48
200^2	2.22×10^{-2}		1.62×10^{-2}	
		1.42		1.16
400^2	8.31×10^{-3}		7.27×10^{-3}	

While difficult to demonstrate using simple advection tests, it is these two features that make the DDR scheme particularly attractive when simulating multi-phase fluid flows where stability of the free surface interface is paramount. An example of such an application is the simulation of volatile droplets interacting with hot solid surfaces [16], where it was found that the DDR VOF advection algorithm produced more accurate and stable solutions than any of the alternative VOF advection techniques.

4. CONCLUSIONS

A new VOF advection algorithm, termed the DDR scheme, has been presented. The algorithm uses a linear piecewise free surface reconstruction method, combined with a unique fully multi-dimensional boundary flux integration technique.

The performance of the DDR algorithm has been compared against a variety of other VOF advection algorithms, using translation, rotation and shear flow advection tests. The DDR algorithm has been found to be more accurate than piecewise linear and flux limited schemes, and to be of comparable accuracy with the original Youngs algorithm.

The real advantage that the DDR scheme has over alternative advection schemes is stability. The DDR scheme generates no fluid 'flotsam', despite the algorithm not including any debris suppressing features, and conserves fluid volume rigorously. While difficult to demonstrate using simple advection tests, these features make the scheme ideal for applications where stability of the free surface interface is paramount.

REFERENCES

1. Nichols BD, Hirt CW, Hotchkiss RS. SOLA-VOF: a solution algorithm for transient fluid flow with multiple free boundaries. Technical Report LA-8355, Los Alamos Scientific Laboratory, 1980.
2. Rider WJ, Kothe DB. Reconstructing volume tracking. *Journal of Computational Physics* 1998; **141**: 112–152.
3. Rudman M. Volume-tracking methods for interfacial flow calculations. *International Journal for Numerical Methods in Fluids* 1997; **24**: 671–691.
4. Kothe DB, Mjolsness RC, Torrey MD. RIPPLE: a computer program for incompressible flows with free surfaces. Technical Report LA-12007-MS, Los Alamos National Laboratory, 1994.
5. Zalesak ST. Fully multidimensional flux-corrected transport algorithms for fluids. *Journal of Computational Physics* 1979; **31**: 335–362.
6. Noh W, Woodward P. SLIC (simple line interface method). *Lecture Notes in Physics* 1976; **59**: 330.
7. Lafaurie B, Nardone C, Scardovelli R, Zaleski S, Zanetti G. Modelling merging and fragmentation in multiphase flows with SURFER. *Journal of Computational Physics* 1994; **113**: 134–147.
8. Chorin AJ. Flame advection and propagation algorithms. *Journal of Computational Physics* 1980; **35**(1): 1–11.
9. Barr PK, Ashurst WT. An interface scheme for turbulent flame propagation. Technical Report SAND82-8773, Sandia National Laboratories, 1984.
10. Debar R. Fundamentals of the KRAKEN code. Technical Report UCIR-760, LLNL, 1974.
11. Youngs DL. Time-dependent multimaterial flow with large fluid distortion. In *Numerical Methods for Fluid Dynamics*, Morton K, Baines M (eds). Academic Press: New York, 1982; 273–285.
12. Ashgriz N, Poo JY. FLAIR: flux line-segment model for advection and interface reconstruction. *Journal of Computational Physics* 1991; **93**: 449–468.
13. Puckett EG, Almgren AS, Bell JB, Marcus DL, Rider WJ. A high-order projection method for tracking fluid interfaces in variable density incompressible flows. *Journal of Computational Physics* 1997; **130**: 269–282.
14. Youngs DL. An interface tracking method for a 3D Eulerian hydrodynamics code. Technical Report 44/92/35, AWRE, 1984.

15. Puckett EG. A volume of fluid interface tracking algorithm with applications to computing shock wave rarefraction. In *Proceedings of the Fourth International Symposium on Computational Fluid Dynamics*, Davis, CA, 9–12 September. University of California at Davis, 1991; 933–938.
16. Harvie DJE. A hydrodynamic and thermodynamic simulation of droplet impacts on hot surfaces. PhD thesis, Department of Mechanical and Mechatronic Engineering, University of Sydney, NSW, Australia, 1999.
17. Harvie DJE, Fletcher DF. A new volume of fluid advection algorithm: the stream scheme. *Journal of Computational Physics* 2000; **162**: 1–32.

