Proc. of the 9th Workshop on Two-Phase Flow Predictions, Merseburg, April 13-16, 1999,
M. Sommerfeld (Ed.), Martin-Luther-Universität Halle-Wittenberg, 2000, pp. 175-184

An Improved Volume of Fluid Method for Numerical Simulation of Clusters of Bubbles

W. Sabisch, M. Wörner, G. Grötzbach, D.G. Cacuci
Forschungszentrum Karlsruhe, Institut für Reaktorsicherheit
Postfach 3640, D-76021 Karlsruhe, Germany

Abstract

A new volume of fluid reconstruction and advection algorithm is presented which correctly reproduces a plane interface regardless of its orientation. The interface is represented by an interpolating plane; their unit normal vector is determined out of tangential vectors at the cell faces. These tangential vectors are obtained straight forward by analytical integrations of local planes defined by the volume fractions of adjacent cells. The algorithm is named EPIRA which is a synonym for Exact Plane Interface Reconstruction and Advection. Numerical tests for different fluid configurations were performed. Results are presented in the paper for spherical drops which are resolved by 8, 16 and 32 mesh cells per diameter, respectively. The encouraging results let us expect to achieve accurate numerical simulations of clusters of bubbles by using 10 - 12 mesh cells per bubble in each direction.

1 Introduction

The phenomenon of gas bubbles rising in a continuous liquid is of significant practical relevance for a variety of engineering applications. The physics of a bubbly flow is - as well as that of any other type of two-phase or multi-fluid flow - to a large extend governed by the interface across which momentum (and in general also heat and mass) are exchanged between the phases. A number of algorithms have been developed for the detailed numerical investigation of physical systems involving interfaces, e.g. front-tracking methods (Univerdi and Tryggvason, 1992), level set methods (Osher and Sethian, 1988), and volume-tracking methods, such as the volume of fluid scheme (Hirt and Nichols, 1981).
All the above methods have their strengths and weaknesses. In the front-tracking method of Univerdi and Tryggvason (1992) the evolution of the interface is accurately described by advecting the front using an unstructured grid that moves through the stationary grid used for the Navier-Stokes solver. However, with this method it is difficult to treat interfaces that interact (e.g. two coalescing bubbles). Volume of fluid methods and level set methods do not require special procedures to model topological changes of the interface. Common to both latter methods is the introduction of a marker function to track the interface. In the level set method this marker function is the signed distance from the interface, whereas in the volume of fluid method it is the volume fraction f of one of the fluids. The advantages and disadvantages of the level set and volume of fluid method are almost reverse. In the volume of fluid method the evolution of the interface is described by solving a conservation law for the volume fraction. A main advantage is therefore that the volume of each fluid is conserved. The level set method does not have the same conservation property. However, an advantage is that there is no need to explicitly reconstruct the interface position from the level set function. In the volume of fluid method the reconstruction of the interface position from the discrete volumetric fractions is a central part. Because of the conservation property and clear physical interpretation of the marker function, in the present study the volume of fluid concept is chosen to develop a code for numerical investigation of single bubbles and clusters of bubbles.

For two-dimensional configurations a number of VOF reconstruction and advection algorithms exist (see the recent review by Rider and Kothe, 1998). With respect to the reconstructed interface geometry, these can be classified in piecewise constant and piecewise linear methods. Two-dimensional piecewise linear methods - like the FLAIR algorithm of Ashgriz and Poo (1991) - reproduce an interface represented by a line correctly. The extension of these methods from two-dimensional to three-dimensional configurations is not straight forward. E.g. the extension of the FLAIR algorithm would result in an exaggerated case check procedure with an enormous computational effort. For this reason a reconstruction algorithm which correctly reproduces a plane interface regardless of its orientations was missing up to now. Perhaps, the most accurate three-dimensional VOF reconstruction algorithm available is the one proposed by Gueyffier et al (1999). The authors, however, find that in general for a planar interface the algorithm does not yield exact results. In the worst case, the accuracy is within 10%.
To minimize the computational costs in a numerical simulation of a cluster of bubbles, it is essential to have an reconstruction and advection algorithm available which yields suffiently accurate results while resolving the diameter of a single bubble by as few mesh cells as possible. Therefore, there is a strong need for an improved three-dimensional VOF reconstruction and advection algorithm.
In the present paper a new algorithm (EPIRA) for three dimensional interface reconstruction and advection on a structured non-equidistant grid consisting of rectilinear mesh cells is presented. EPIRA is a synonym for <u>E</u>xact <u>P</u>lane <u>I</u>nterface <u>R</u>econstruction and <u>A</u>dvection. EPIRA avoids case checks by calculating the unit normal vector out of tangential vectors at the cell faces. These tangential vectors are straight forward and numerical efficiently obtained by analytical integrations of local planes defined by the volume fractions of adjacent cells. With the EPIRA algorithm a plane interface of arbitrary orientation is always reconstructed and advected in an exact manner. The EPIRA reconstruction and advection steps are presented in chapter 2. In chapter 3 test cases for a drop in a uniform velocity field are presented. The paper is closed by the conclusions and outlook, where the next step of code development, i.e. the coupling of EPIRA with a Navier-Stokes solver, is outlined.

2 EPIRA algorithm

2.1 Computational grid

For the discretisation of the flow domain we use a structured rectilinear grid with mesh cell centers located at $\mathbf{x}_{i,j,k} = (x_i, y_j, z_k)^T$ and mesh cell faces located at $x_{i\pm1/2} = x_i \pm \Delta x_i/2$, $y_{j\pm1/2} = y_j \pm \Delta y_j/2$ and $z_{k\pm1/2} = z_k \pm \Delta z_k/2$. The corresponding mesh width Δx_i, Δy_j and Δz_k may vary. For brevity of nomenclature, we denote the six faces of cell (i, j, k) by E(ast) $(x_{i+1/2})$, W(est) $(x_{i-1/2})$, N(orth) $(z_{k+1/2})$, S(outh) $(z_{k-1/2})$, B(ack) $(y_{j+1/2})$, F(ront) $(y_{j-1/2})$ and use $E_{i,j,k}$ as abreviation to denote face East of cell (i, j, k). Within each mesh cell, the liquid volumetric fraction $f_{i,j,k}$ is defined as the volume occupied by the liquid phase divided by the total volume $V_{i,j,k}$ of the cell. Thus, in a pure gas cell it is $f = 0$, in a pure liquid cell it is $f = 1$ and in an interface mesh cell it is $0 < f < 1$.

2.2 Reconstruction

To reconstuct the interface from the discrete liquid volumetric fractions $f_{i,j,k}$, we assume a functional interface of zero thickness which can locally be described by a single valued height function $z = h(x, y)$, where x, y, z define a local Carthesian co-ordinate system. From a Taylor expansion of $h(x, y)$ around (x_0, y_0) we obtain

$$\begin{aligned} h(x,y) &= h(x_0,y_0) + (x-x_0)\,\frac{\partial}{\partial x}h(x,y)\Big|_{(x_0,y_0)} + (y-y_0)\,\frac{\partial}{\partial y}h(x,y)\Big|_{(x_0,y_0)} \\ &\quad +(x-x_0)(y-y_0)\,\frac{\partial^2}{\partial x \partial y}h(x,y)\Big|_{(x_0,y_0)} + \cdots \\ &= h_0 + \alpha(x-x_0) + \beta(y-y_0) + HOT \qquad (1) \end{aligned}$$

In the EPIRA algorithm it is assumed that the interface can locally be approximated by a plane, i.e. by the first three terms on the r.h.s. of Eq. (1). In the vicinity of h_0 this is always a good approximation.
A plane is completely defined by a normal vector $\mathbf{n} = (n_1, n_2, n_3)^T$ and a point $\mathbf{b} = (b_1, b_2, b_3)^T$ lying within the plane. In the following we describe how within the EPIRA algorithm $\mathbf{n}_{i,j,k}$ and $\mathbf{b}_{i,j,k}$ are determined for each interface cell from the discrete values $f_{i,j,k}$. This is done in three main steps. In the first one, tangential vectors $\mathbf{t}$ are computed for each face of a mesh cell. In the second step, from these tangential vectors one normal vector, which is representative for the mesh cell center, is determined. Finaly, $\mathbf{b}_{i,j,k}$ is determined in such a way, that the planar interface divides the computational cell into two parts containing the adequate volume of each fluid.

2.2.1 Determination of face-centered tangential vectors

For determining the tangential vector $\mathbf{t}$ at a certain cell face, the corresponding neighbouring cell needs to be also considered. If this neighbouring mesh cell is not an interface cell (i.e. $f = 0$ or $f = 1$) the interface does not cut this face of the mesh cell, and $\mathbf{t}$ is set to zero. Otherwise, this face is called an interface-face and $\mathbf{t}$ is non-zero and must be computed. As an example, this is demonstrated here for $\mathbf{t}_E$ and the configuration sketched in Figure 1.

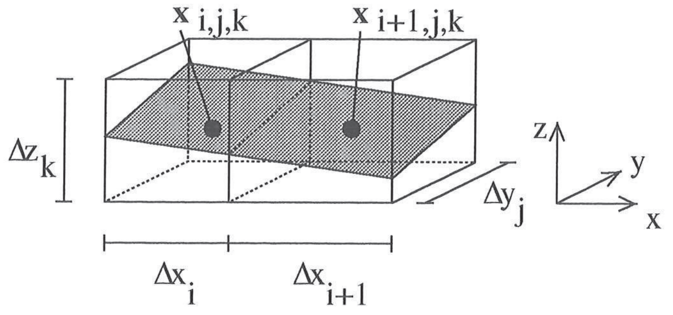

Figure 1: Cells (i, j, k) and $(i + 1, j, k)$ with plane $h(x, y)$ representing the interface. The fluid is below $h(x, y)$.

In cases where the function $h(x, y)$ does not cut the upper or lower faces of the two adjacent cells the relations

$$f_{i,j,k} = \frac{1}{\Delta x_i\, \Delta y_j\, \Delta z_k} \int_0^{\Delta x_i} \int_0^{\Delta y_j} h(x,y)dy\,dx \qquad (2)$$

$$f_{i+1,j,k} = \frac{1}{\Delta x_{i+1}\, \Delta y_j\, \Delta z_k} \int_{\Delta x_i}^{(\Delta x_i+\Delta x_{i+1})} \int_0^{\Delta y_j} h(x,y)dy\,dx \qquad (3)$$

hold (origin of co-ordinate system is at $(x_{i-1/2}, y_{j-1/2}, z_{k-1/2})$ and $(x_0, y_0) = (\Delta x_i, \Delta y_j/2))$. By inserting the linear function $h(x, y)$ of Eq. (1) in Eqs. (2) and (3), performing the integration analytically and subtracting both equations one obtains after some rearrangement

$$\alpha_{i+1/2} = \frac{\partial}{\partial x} h(x,y)\Big|_{(\Delta x_i, \Delta y_j/2)} = \frac{2\Delta z_k}{\Delta x_i + \Delta x_{i+1}} (f_{i+1,j,k} - f_{i,j,k}) \tag{4}$$

Since $\alpha_{i+1/2}$ represents the gradient of $h(x,y)$ in x-direction, we obtain for the unit tangential vector $\mathbf{t}_E$, lying here in plane x-z, the result

$$\mathbf{t}_E = \frac{1}{\sqrt{1+\alpha_{i+1/2}^2}} \begin{pmatrix} 1, & 0, & \alpha_{i+1/2} \end{pmatrix}^T \tag{5}$$

Note, that for any planar interface this tangential vector is exact, if the supposed lower and upper integration limits in Eqs. (2) and (3) are correct. This requires that the interface does not cut the upper or lower faces of both neighbouring cells (i.e. $N_{i,j,k}$, $N_{i+1,j,k}$, $S_{i,j,k}$, $S_{i+1,j,k}$ are not interface-faces). If this is not the case, to be still exact the integration limits in Eqs. (2) and (3) must be adapted to account for the cut of the interface with the lower and upper cell faces. This modification would result in a number of different cases to determine the adequate integration limits. To avoid this case distinction, here we use another approach.
Let us assume that the interface in cell (i,j,k) cuts the upper face, i.e. face $N_{i+1,j,k}$ is an interface-face (see Figure 2, left). To avoid the adaptation of the integration limits which would be necessary to obtain an exact tangential vector, the idea is simply to modify the cell volume used for determining $\mathbf{t}$. This is done by an extension in the adequate direction. For the example displayed in Figure 2 we now consider two extended neighbouring cells, consisting of cells $(i,j,k)+(i,j,k+1)$ and $(i+1,j,k)+(i+1,j,k+1)$, respectively (see Figure 2, right). Due to this extension, now Eqs. (2) and (3) can again be used to calculate $\mathbf{t}_E$ exactly, when the left hand sides are replaced by $f_{i,j,k}+f_{i,j,k+1}$ and $f_{i+1,j,k}+f_{i+1,j,k+1}$, respectively, and on the r.h.s. Δz_k is replaced by $\Delta z_k + \Delta z_{k+1}$. An equivalent procedure is applied in cases where the interface cuts the lower faces of the cells.
If one or both of the slopes α and β (see Eq. (1)) are very steep, one upper or lower extension might not be sufficient to obtain $\mathbf{t}$ exactly. Because we do not want to lose the locality of the reconstruction, not more than one extension above and below the basic pair of cells is used. If this is not sufficient (e.g. if face $N_{i,j,k+1}$ in Figure 2 would be an interface-face), the 2D FLAIR algorithm (Ashgriz and Poo, 1991), adapted to non-equidistant grids, is applied to get a guess for the slope. Thus, the interface slope β is assumed to be zero and the problem is simplified to a two-dimensional configuration. Then, by considering four possible different cases (see Figure 3) the slope α is determined analytically via a case check diagram according to the 2D FLAIR algorithm. From this value for α, a guess for $\mathbf{t}$ is determined from Eq. (5).

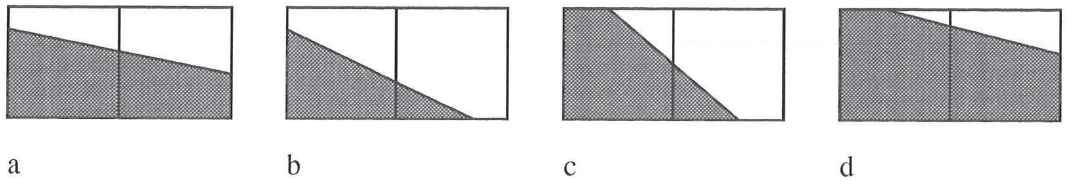

Figure 3: Four cases of 2D FLAIR algorithm (Ashgriz and Poo, 1991).

By the above procedure, at each of the minimum three and maximum six interface-faces of an interface mesh cell a unit tangential vector $\mathbf{t}$ is computed. The tangential vector is flagged as 'exact' if the slope is determined via Eq. (4) and is flagged as 'non-exact' if the 2D FLAIR algorithm is used instead.

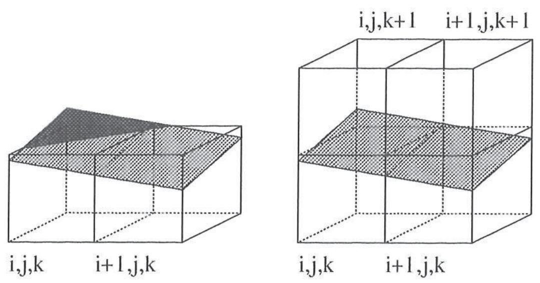

Figure 2: Example where the interface cuts the upper face of the cells (left). An extension of the left and right volume used to compute the slope at face E is performed (right).

2.2.2 Determination of cell-centered unit normal vector

From the tangential vectors $\mathbf{t}$ at the different faces of cell (i, j, k) a cell-centered unit normal vector $\mathbf{n}$ is computed for each interface cell. For illustration, this is demonstrated here for the example sketched in Figure 4. For this situation, $\mathbf{t}_N$ and $\mathbf{t}_S$ are zero, whereas $\mathbf{t}_E$, $\mathbf{t}_W$, $\mathbf{t}_F$ and $\mathbf{t}_B$ are non-zero and either flagged exact or not. At first, for each co-ordinate direction one representative tangential vector is determined. If e.g. $\mathbf{t}_E$ and $\mathbf{t}_W$ lie within the same plane and obey the same flag (i.e. exact or non-exact) a simple average

$$\mathbf{t}_{EW} = \frac{1}{2}(\mathbf{t}_E + \mathbf{t}_W)$$

and subsequent normalisation is performed. Otherwise, either $\mathbf{t}_E$ or $\mathbf{t}_W$ is selected as representative by an appropriate procedure, where exact flagged tangential vectors are preferred. In an equivalent way, $\mathbf{t}_{NS}$ and $\mathbf{t}_{FB}$ are determined. From these three representative tangential vectors three different preliminary normal vectors are determined

$$\mathbf{n}_1 = \mathbf{t}_{EW} \times \mathbf{t}_{NS}, \qquad \mathbf{n}_2 = \mathbf{t}_{NS} \times \mathbf{t}_{FB}, \qquad \mathbf{n}_3 = \mathbf{t}_{FB} \times \mathbf{t}_{EW}, \tag{6}$$

by computing the vector product of each possible pair. If necessary, a normal vector is re-orientated so as to point inside the fluid. Eventually, the cell-centered unit normal vector is computed by averaging of $\mathbf{n}_1, \mathbf{n}_2, \mathbf{n}_3$, where again 'exact' vectors are preferred, and is finally normalized. For the situation of Figure 4 e.g., $\mathbf{t}_{NS}$ is zero. Therefore, only the normal vector $\mathbf{n}_3$ is non-zero and no averaging with $\mathbf{n}_1$ and $\mathbf{n}_2$ is performed. From the unit normal vector $\mathbf{n} = (n_1, n_2, n_3)^T$ the slopes $\alpha = -n_1/n_3$ and $\beta = -n_2/n_3$ may easily be computed.

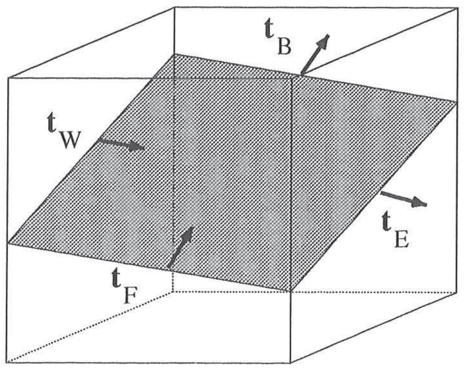

Figure 4: Interface cell with four tangential vectors $\mathbf{t}_E$, $\mathbf{t}_W$, $\mathbf{t}_B$ and $\mathbf{t}_S$.

2.2.3 Determination of point b

The axis element $\mathbf{b}_{i,j,k}$ for each interface cell is determined by an iterative procedure. Because $h := h(\mathbf{n}_{i,j,k}, \mathbf{b}_{i,j,k})$ we can write

$$f_{i,j,k} = \frac{1}{V} \int h(\mathbf{n}_{i,j,k}, \mathbf{b}_{i,j,k})\, dV \tag{7}$$

Both, $f_{i,j,k}$ and $\mathbf{n}_{i,j,k}$ are known. So $\mathbf{b}_{i,j,k}$ is iteratively determined until Eq. (7) is fulfilled. In practise, in most of the interface cells convergence is reached within only 2 to 4 iteration steps. To compute the integral in equation (7) the same procedure is used as for the advection step, which will be explained next.

2.3 Advection

In the advection step the fluxes of the liquid phase across the faces of the cells are computed. The algorithm used within EPIRA is based on an operator-splitting, i.e. for each direction a separate sweep is performed. In the following, we illustrate the procedure for the sweep in x-direction; an interface configuration as displayed in Figure 5. With $u_x > 0$ as the velocity component normal to face E(ast), defined at $(x_{i+1/2}, y_j, z_k)$, within a time step Δt the shaded volume is fluxed across face E into the neighbouring cell. To calculate the flux δf, e.g. in the 2D FLAIR algorithm a large number of different cases are considered. For three dimensions even more complicated cases would occur. To avoid complicated case distinctions as far as possible, within EPIRA a quite general algorithm for calculating δf is developed.

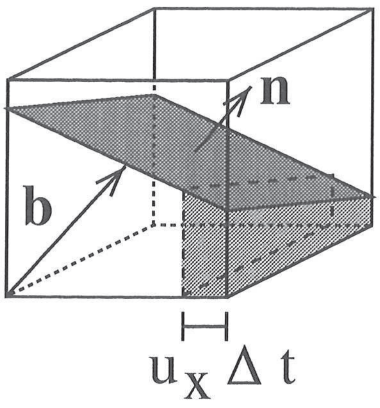

Figure 5: Within Δt the light shaded volume is advected across face E(ast).

For the linear height function

$$h(x,y) = (b_1 - x)\frac{n_1}{n_3} + (b_2 - y)\frac{n_2}{n_3} + b_3 \tag{8}$$

representing the interface in the cell without loss of generality we can assume $n_1 > 0$, $n_2 > 0$ and $n_3 > 0$. If not all components of the unit normal vector are positive, the co-ordinate system is rotated such that in the new system $\mathbf{n} = (n_1, n_2, n_3)^T$ has only positive components. We define a volume integration operator V

$$V(x_1, x_2, y, z) := \int_{x_1}^{x_2} \int_{y}^{y_m(x)} h_m(x', y')\, dy'\, dx' \tag{9}$$

with

$$y_m(x) := \min\{\max\{y, y(x)\}, M\}; \qquad h_m(x,y) := \min\{\max\{0, h(x,y) - z\}, M\} \tag{10}$$

and

$$y(x) := \left[(b_1 - x)\frac{n_1}{n_3} + b_3 - z \right] \frac{n_3}{n_2} + b_2. \tag{11}$$

The integrals for calculating V can be evaluated analytically. The volume integral operator V gives the volume under the function $h(x, y)$ in the domain $[x_1, x_2] \times [y, y_m] \times [z, h_m]$. In Eq. (10) M is a positive limit which is only needed in cases where $n_1 = 0$ or $n_2 = 0$. In our calculations we use $M = 1000$. By combining computed values of V in appropriate different domains, it is possible to calculate the volume flux δV in a specific volume $[x_l, x_r] \times [y_l, y_r] \times [z_l, z_r]$ via the relation

$$\delta V = V(x_l, x_r, y_l, z_l) - V(x_l, x_r, y_r, z_l) - V(x_l, x_r, y_l, z_r) + V(x_l, x_r, y_r, z_r), \tag{12}$$

see Figure 6. The flux of the liquid volumetric fraction is then simply determined from $\delta f = \delta V/(\Delta x_i \Delta y \Delta z)$.

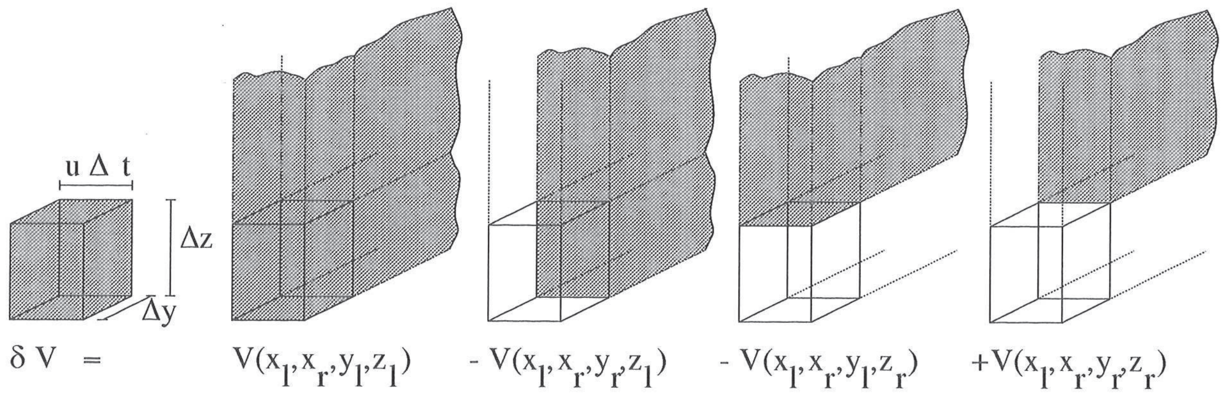

Figure 6: Calculation of the volume flux δV. The integration domain is marked by the shaded surface.

To calculate the fluxes of the volume fraction in a general co-ordinate direction for a positive velocity the integration limits given in Table 1 have to be used.

u	x_l	x_r	y_l	y_r	z_l	z_r
$u_x > 0$	$x_{i+1/2} - u_x \Delta t$	$x_{i+1/2}$	$y_{j-1/2}$	$y_{j+1/2}$	$z_{k-1/2}$	$z_{k+1/2}$
$u_y > 0$	$x_{i+1/2}$	$x_{i+1/2}$	$y_{j+1/2} - u_y \Delta t$	$y_{j+1/2}$	$z_{k-1/2}$	$z_{k+1/2}$
$u_z > 0$	$x_{i+1/2}$	$x_{i+1/2}$	$y_{j-1/2}$	$y_{j+1/2}$	$z_{k+1/2} - u_z \Delta t$	$z_{k+1/2}$

Table 1: Integration limits in the EPIRA advection step to compute the flux δV in the different co-ordinate directions.

3 Test Cases

To test the performance of the EPIRA algorithm several three-dimensional test cases have been performed. Some results of special interest are presented in this chapter.

3.1 Test of reconstruction algorithm

For testing of the reconstruction algorithm for a certain 'frozen' interface topology first the discrete field of liquid volumetric fractions $f_{i,j,k}$ in the computational domain is generated

by a separate initialisation job. The field $f_{i,j,k}$ is given as input to the EPIRA reconstruction step and the interface position $(\mathbf{n}, \mathbf{b})$ is reconstructed by the algorithm described in section 2.2.
Results for the test cases performed for plane interfaces of different orientation will not be presented here. We just note, that these are always reconstructed correctly. This was tested for an equidistant grid with isotropic mesh width and for a grid which is non-equidistant in one direction. The exact reconstruction of the normal vector of a plane interface can be attributed to the extension of the cells considered in determining the interface slope at cell faces. Due to this procedure, for a plane interface one obtains always at least two independent exact face-centered tangential vectors, from which the correct normal vector is obtained by the vector product.
In the following we present test cases performed for a spherical drop. We take a drop instead of a bubble simply because of the easier and more instructive visualisation when f is defined as liquid volumetric fraction. For discretisation, an equidistant grid with isotropic mesh width is used. Results are given for three different simulations where the drop diameter is resolved by 8, 16, and 32 mesh cells, respectively. In Table 2 the L_1 error of the EPIRA reconstruction is given for the different resolutions. This is computed by comparing the reconstructed unit normal vector within each interface mesh cell with the exact one. In Table 2 the number of interface mesh cells N_{int} is given, as well as the order of the reconstruction scheme, which is computed from the L_1 errors. As can be expected, for this full three-dimensional problem the reconstruction step is first order accurate.

drop resolution	$\Delta x = \Delta y = \Delta z$	N_{int}	$L_1(t=0)$	order	N_{time}	$\varepsilon_{\lvert\Delta f\rvert}$
$8 \times 8 \times 8$	0.1	272	0.0417		160	0.048
				1.17		
$16 \times 16 \times 16$	0.05	1160	0.0185		320	0.017
				1.06		
$32 \times 32 \times 32$	0.025	4760	0.0089		640	0.00032

Table 2: Three test cases of spherical drops

3.2 Test of advection algorithm

A test of the pure EPIRA advection algorithm is not meaningful because in practise always the interface first must be reconstructed before it can be advected. Therefore, we now test the complete EPIRA algorithm. The test computations are performed in a cubical domain with periodic boundary conditions in $x-$ and $y-$direction. In $z-$direction the computational domain is confined by rigid walls. In all the calculations a uniform velocity field with the walls moving according to this velocity is used. Thus, any deformation of the initial interface geometry during the computation can be directly attributed to inaccuracies of the EPIRA algorithm.
In all computations, the conservation of the liquid volumetric fraction is fulfilled exactly within a certain user defined limit ϵ. After each time step, in cells where $0 < f < \epsilon$ the liquid volumetric fraction is set to zero, while in cells with $1 - \epsilon < f < 1$ it is set to unity. In our test computations we used $\epsilon = 10^{-8}$.
Again we show no results for plane interfaces. We note, however, that in the test computations a layer of fluid lying between two parallel planar interfaces of arbitrary orientation is always advected exactly.

In the following we present results for a single spherical drop in a uniform velocity field $\mathbf{u} = (u_1, u_2, u_3)^T = (5, 0, 0)^T$. In all computations, the time step width Δt corresponds to a Courant number $C = (u_1 \Delta t)/\Delta x = 0.5$. For each case a downstream transport of 10 drop diameters is simulated. This corresponds to a problem time of $t = 1.6$, which is computed within N_{time} time steps, see Table 2. Due to the periodic boundary conditions the position of the transported drop within the computational domain should then exactly agree with its initial position. To quantify the error of the advection the quantity

$$\varepsilon_{|\Delta f|} = \frac{\sum_{i,j} |f_{i,j,k}(t = 1.6) - f_{i,j,k}(t = 0)|}{\sum_{i,j} f_{i,j,k}(t = 0)} \tag{13}$$

is computed for the horizontal mid-plane of the drop and given in Table 2. For an absolutely exact advection algorithm $\varepsilon_{|\Delta f|}$ is zero, while in the worst case where the final and initial bubble positions do not overlap at all, $\varepsilon_{|\Delta f|}$ is unity. For the EPIRA algorithm even for the coarse grid $\varepsilon_{|\Delta f|}$ is below 5% and below 0.04% for the fine grid.
In Figure 7 and 8 the horizontal mid-planes of the drops for the coarse and medium drop resolution are shown.

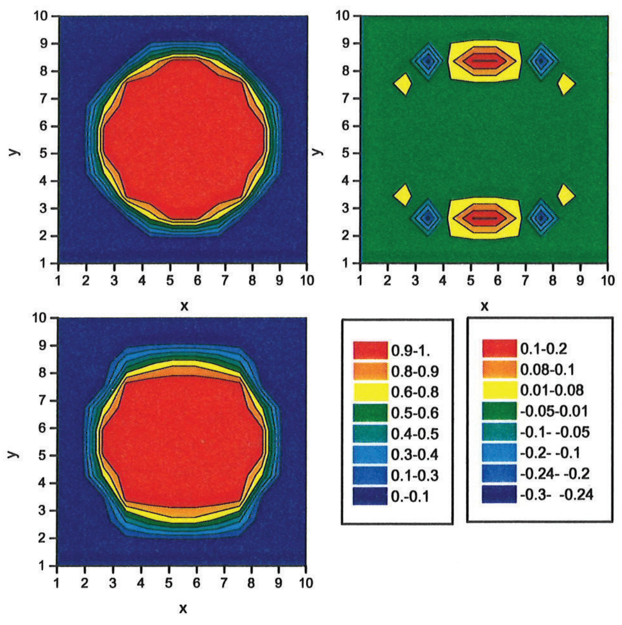

Figure 7: Plot of initial and final f distribution in mid-plane for drop, resolved by 8 mesh cells per diameter, and computational error.

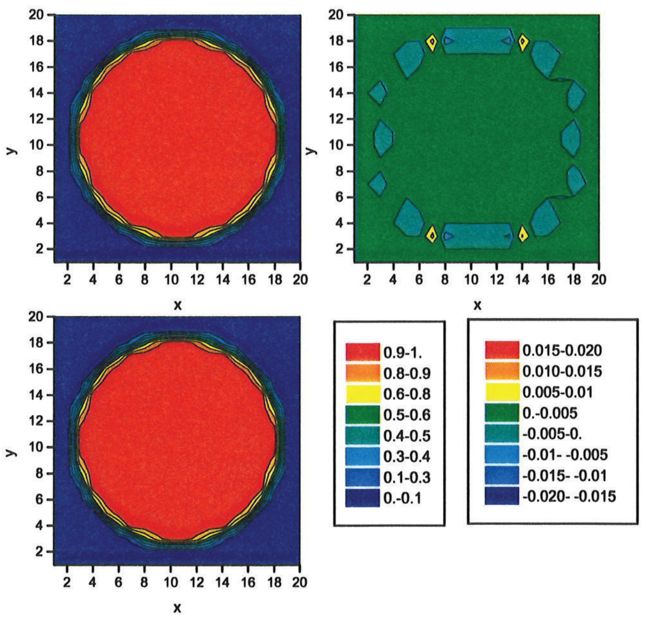

Figure 8: Plot of initial and final f distribution in mid-plane for drop, resolved by 16 mesh cells per diameter, and computational error.

In the upper left corner of Figs. 7 and 8 the initial liquid volumetric fraction is displayed, while in the lower left corner it is given for $t = 1.6$ (left color legend applies). In the upper right corner the differences between $t = 1.6$ and $t = 0$ are visualised (right color legend). Due to lack of space, the results for the fine mesh are not shown here.
As can be seen in Figure 7, for the coarse grid we get a deformation of the bubble after N_{time} time steps, although the sum of all $f_{i,j,k}$ in each horizontal plane is conserved. The error is accumulated at that sides of the drop where the unit normal vectors are perpendicular to the main flow direction. For the medium and fine resolutions no such deformation is visible after $t = 1.6$ and the small error is almost uniformly distributed.

4 Conclusions and outlook

A new algorithm is presented for the three-dimensional volume of fluid interface reconstruction and advection on an orthogonal fixed Eulerian grid. This algorithm, called EPIRA, reconstructs and advects a plane interface in an exact manner, regardless of its orientation. Thus, it is accurate to first order. In both, the reconstruction and the advection step, it is intended to avoid complicated case distinctions as far as possible. Instead a formal deduction of the unit normal vector of the reconstructed interface is used.
Test computations for planes and single drops in a uniform velocity field confirm the excellent volume conservation, which was expected from a volume of fluid method. The advection step of EPIRA imposes no new restrictions for the time step width Δt. In the test computations a Δt corresponding to a Courant number of 0.5 was used and a problem time which corresponds to the downstream transport of the drop by ten diameters was simulated.
For the coarse resolution, where the drop diameter is resolved by only 8 mesh cells, a distinct deformation of the drop is observed after 160 time steps. Nevertheless, the EPIRA algorithm yields surprisingly accurate results. With the medium resolution of 16 cells per diameter no such deformation is observed and the algorithm performs excellently. The same holds for the fine resolution of 32 mesh cells per diameter. It appears that for practical simulations a resolution of $10-12$ mesh cells per diameter may be sufficient. As the use of a grid consisting of about $100 \times 100 \times 100$ mesh cells seems possible with todays computer power, the EPIRA algorithm will certainly allow for accurate simulations of a cluster of bubbles.
As the next step of code development, the coupling of the EPIRA algorithm with a Navier-Stokes solver will be completed. Within the latter, the continuity and momentum equations formulated for the mixture density and center of mass velocity of the two-phase system are solved by a projection method. The convective term in the mixture momentum equation is discretised by an Essentially-Non-Oscillatory scheme, in order to avoid numerical smearing of the interface within the three-step TVD Runge-Kutta time integration. The surface tension force is represented as a volume force according to the Continuum Surface Force (CSF) concept of Brackbill et al. (1992). While in the standard CSF model the interface unit normal vector is approximated within the surface tension term by the gradient of the discrete f values, in the present work the exact normal vectors obtained from the EPIRA reconstruction will be utilised.

5 Literature

Ashgriz N., Poo J.Y., 1991: FLAIR - flux line-segment model for advection and interface reconstruction, J. Comp. Phys., 93, 449-468.
Brackbill J.U., Kothe D.B., Zemach C., 1992: A continuum method for modeling surface tension; J. Comp. Phys., 100, 335-354.
Gueyffier D., Li J., Nadim A., Scardovelli R., Zaleski S., 1999: Volume-of-Fluid interface tracking with smoothed surface stress methods for three-dimensional flows; to appear in J. Comp. Phys.
Hirt C.W., Nichols B.D., 1981: Volume of fluid (VOF) method for the dynamics of free boundaries; J. Comp. Phys., 39, 201-225.
Osher S., Sethian J.A., 1988: Fronts propagating with curvature dependent speed: Algorithms based on a Hamilton-Jacobi formulation, J. Comp. Phys., 79, 12-49.
Rider W.J., Kothe D.B., 1998: Reconstructing volume tracking; J. Comp. Phys., 141, 112-152.
Univerdi S.O., Tryggvason G., 1992: A front-tracking method for viscous, incompressible, multi-fluid flows; J. Comp. Phys., 100, 25-37.