Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Computational Physics 188 (2003) 611–639
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

A mixed markers and volume-of-fluid method for the reconstruction and advection of interfaces in two-phase and free-boundary flows

Eugenio Aulisa, Sandro Manservisi, Ruben Scardovelli *

INFM-BO and DIENCA, University of Bologna, Lab. di Montecuccolino, Via dei Colli 16, Bologna 40136, Italy

Received 11 July 2002; received in revised form 19 March 2003; accepted 19 March 2003

Abstract

In this work we present a new mixed markers and volume-of-fluid (VOF) algorithm for the reconstruction and advection of interfaces in the two-dimensional space. The interface is described by using both the volume fraction function C, as in VOF methods, and surface markers, which locate the interface within the computational cells. The C field and the markers are advected by following the streamlines. New markers are determined by computing the intersections of the advected interface with the grid lines, then other markers are added inside each cut cell to conserve the volume fraction C. A smooth motion of the interface is obtained, typical of the marker approach, with a good volume conservation, as in standard VOF methods. In this article we consider a few typical two-dimensional tests and compare the results of the mixed algorithm with those obtained with VOF methods. Translations, rotations and vortex tests are performed showing that many problems of the VOF technique can be solved and a good accuracy in the geometrical motion and mass conservation can be achieved.

PACS: 65C20; 68U20

Keywords: Two-phase flow; Reconstruction and advection of interfaces; Hybrid method; Front tracking; Volume tracking

1. Introduction

Various versions of volume-of-fluid (VOF) or volume tracking methods have been used successfully for the numerical simulation of two-phase and free-surface flows with implicit tracking of the interface. In the past several years many algorithms have been devised to track and advect the interface with a formulation varying from algebraic to geometric or even heuristic. Several reviews of early and present VOF algorithms have been written [24–26], with an extensive bibliography as well. The most recent technique is the

* Corresponding author. Tel.: +39-051-644-1720; fax: +39-051-644-1747.
E-mail address: raus@mail.ing.unibo.it (R. Scardovelli).

0021-9991/03/$ - see front matter
doi:10.1016/S0021-9991(03)00196-7

piecewise-linear interface calculation (PLIC) [20,24,39], which is mainly geometric in nature. It is based on a linear approximation of the interface in each computational cell, i.e., a segment in two dimensions and a portion of a plane in three dimensions, and the overall reconstruction is piecewise-linear. The algorithms based on this method perform well with smooth interfaces with a local radius of curvature larger than the grid spacing h, otherwise merging or breaking is performed with no accuracy, generating sometimes incorrect topologies. In particular, in the presence of thin filaments the method breaks the interface in numerous sub-grid structures whose motion cannot be correctly solved. In VOF methods the two phases are usually differentiated by a scalar function C (also known as *color function*), which represents the fraction of each grid cell occupied by one of the two fluids, assumed as the reference phase. The function C varies monotonically between the constant value one, in the full cells, and zero, in the empty cells, while the interface is localized in the transition region. For the interface reconstruction or numerical simulations of flows with interfaces possessing surface tension [4,38] or in touch with walls along contact lines [21], it is required the estimate of a few geometrical quantities such as the unit normal $\vec{n}$ and the mean curvature κ. These geometrical quantities are usually computed using numerical spatial derivatives of the color function C or of a smoothed color function $\tilde{C}$. However, the interface reconstruction is not continuous at the cell boundary and the discontinuity becomes large with high curvature [28]. In these conditions the calculation of the normal vector and the local curvature is usually rather crude even when smoothing techniques are applied. The interface is then tracked in time with a geometric evaluation of the volume fluxes across the cell boundary. This can be done independently along each coordinate direction, with multidimensionality obtained via an operator split technique [8,24,25,28], or with multidimensional unsplit schemes that are more accurate but geometrically more complex [10,11,17,24].

Other popular Eulerian front-capturing schemes, able to capture or define in some way an interface, include the level set [16,29] and phase-field approaches [12,13]. They have been successfully used to simulate merging, breaking and large topology changes.

With Lagrangian methods, such as front-tracking schemes [7,18,36,37] and marker particles [5,9,22,35], the reconstruction is fairly mesh independent and the determination of the interface position simpler. The interface is smooth and the computation of quantities such as the normal vector and the curvature is definitely more reliable. In spite of this, conservation of mass may not be very accurate and sometimes topology changes, which for front-tracking may require addition or deletion of interface elements, cannot be handled in an easy way, particularly in three-dimensional flows.

Several incompressible test problems have been proposed to compare how well Lagrangian and Eulerian tracking schemes describe fluid bodies that in large vortical flows develop filamentary interface structures [22–24]. As the fluid filaments are progressively stretched by the flow, they become too thin, of the order of the grid size, to be correctly resolved on a fixed mesh. In this situation VOF methods enforce mass conservation but break filaments in several droplets and level set schemes loose continuously mass deteriorating quickly the front representation. Attempts to improve mass conservation and accuracy have led to a variety of reinizalization schemes [32,33]. For these vortical flows Lagrangian methods are usually more accurate and maintain filamentary structures better than Eulerian methods.

A number of hybrid methods have also appeared in recent years. In [34] a mixed method is proposed that combines the accurate mass conservation of VOF and a better representation of the surface curvature via finite differences of the level set function. The combined method remains Eulerian and the interface description is still not accurate in regions with thin filaments, where a representation of the front with Lagrangian particles would be more appropriate. The method described in [6] combines Lagrangian marker particles and an Eulerian level set method. Marker particles are randomly positioned near the interface (the zero level set) and are passively advected by the flow. The particles are used to rebuild the level set function in under-resolved regions, as in the case of filamentary fluid structures where the level set method suffers from excessive regularization, and to improve its mass conservation properties. A simplified front-tracking method for three-dimensional flows has been recently developed in [30], coupled with a new level contour

reconstruction technique and a global mass conservation algorithm. Interface elements in this representation are physically but not logically connected. As a result, the interface is continuous and there is no need to bookkeep connections between neighboring surface elements. An indicator function is used to reinitialize the front, with the constraint that the total volume enclosed by the new surface is the same as the value before the reconstruction. Effects on local mass conservation are usually negligible, except near high curvature regions.

In this work we present a new hybrid method which uses both markers, to reconstruct and move the interface, and the color function C, to conserve volumes. The interface is described by a continuous chain of segments connecting two types of markers: grid intersection and mass conservation markers, respectively. Intersection markers locate the interface on the grid lines, while conservation markers are added on the interface inside each cell to keep the volume fraction of the reference phase equal to the local value of the color function C. Both types of markers are advected numerically along streamlines to get new intersection markers and to update the scalar field C. The introduction of intersection markers (which were also used in [30] to ensure continuity in the reconstruction of the front and at the same time to automatically model merging and breakup of interfaces) eliminates the need to remesh the system, while the conservation markers are needed to improve the mass conservation properties. With this fully multidimensional technique we obtain a smooth motion of the interface, typical of marker methods, and a good volume conservation, as in standard VOF methods. This work improves both the accuracy of interface tracking, when compared to standard VOF methods, and the conservation of mass, with respect to the original marker method. In this paper we discuss only the two-dimensional algorithm with qualitative and quantitative arguments, since the three-dimensional version will be discussed in a future paper.

The plan of this paper is as follows. In Section 2 we briefly introduce the advection equations for the VOF part of the algorithm and for the markers. Then in Section 3 we describe in detail the discrete formulation of the mixed markers and VOF algorithm. In Section 4 we discuss the results obtained with our formulation for a few standard tests and compare them with the results of VOF methods. In particular, we consider translations, rotations and vortical flows and show that many of the classical weaknesses of VOF methods can be solved, achieving a good accuracy of the interface motion together with mass conservation. Finally, we present our conclusions.

2. Continuous convection equations

2.1. The convection equation for the phase indicator function χ

Let Ω be a bounded domain with the reference phase 1 contained in the subdomain $\Omega_1 \subset \Omega$. Let $\vec{u}$ be a flow field and χ an indicator function for the reference phase defined as

$$\chi(t,\vec{x}) = \int_{\Omega_1} \delta(\vec{x}-\vec{x}')\,\mathrm{d}\vec{x}'. \tag{1}$$

The integral is over the volume $\Omega_1(\vec{x},t)$ bounded by the interface $S(\vec{x},t)$, supposed to be continuous. The distribution $\delta(\vec{x}-\vec{x}')$ is the Dirac delta function that is non-zero only where $\vec{x}'=\vec{x}$. Clearly, the indicator χ is one for all $\vec{x}\in\Omega_1$ and zero on $\Omega-\Omega_1$. We say that the indicator function χ is singular over the set O of positive measure if χ is not zero only for zero measure subsets of O. From the phase indicator we can determine geometric information such as the unit normal $\vec{n}$ and the curvature κ. In particular, for the computation of the interface normal $\vec{n}$ we can take the gradient of (1) and transform the volume integral into an integral over the interface to get

$$\vec{\nabla}\chi(\vec{x},t) = -\int_S \vec{n}(\vec{x}',t)\delta(\vec{x}-\vec{x}')\,\mathrm{d}\vec{s}'. \tag{2}$$

The unit normal $\vec{n}$ to the interface S is defined to point into the reference phase. We note that $\vec{\nabla}\chi$ is a singular function over Ω in the sense defined above, i.e., it is different from zero only over sets of zero measure.

Since the fluid type does not change following the fluid paths, the indicator function χ behaves like a passive scalar and satisfies the following advection equation:

$$\frac{\mathrm{d}\chi}{\mathrm{d}t} = 0,$$

moreover if $\vec{u}$ is a divergence-free flow field

$$\frac{\partial\chi}{\partial t} + \nabla\cdot(\vec{u}\chi) = 0. \tag{3}$$

Here we use a weak formulation of this partial differential equation, since derivatives of the discontinuous function χ are singular. In the weak formulation, the differential equation is interpreted by space integrals, which are well-defined. Therefore, it is more appropriate to integrate (3) over a control volume V

$$\frac{\partial}{\partial t}\int_V \chi\,\mathrm{d}\vec{x} + \int_{\partial V} \vec{n}\cdot(\vec{u}\chi)\,\mathrm{d}\vec{s} = 0 \quad \forall V \subset \Omega, \tag{4}$$

where ∂V is the surface around the control volume V and $\vec{n}$ its normal.

2.2. The equation of motion for the markers

Traditional marker particle schemes place particles, having an identity or *color*, in the whole domain (see for example [9,22]). More recently some authors have proposed methods that require either particles in the neighborhood of the free surface or on the interface (as in [2,6] and [5], respectively). In our mixed markers and VOF method we are interested only in the evolution of the interface $S(\vec{x},t)$, then given a flow field $\vec{u}(\vec{x},t)$, if $\vec{x}\in S$ we have

$$\frac{\mathrm{d}\vec{x}}{\mathrm{d}t} = \vec{u}. \tag{5}$$

The previous equation can be integrated as

$$\vec{x} = \vec{x}_0 + \int_{t_0}^{t} \vec{u}(\vec{x}(t'),t')\,\mathrm{d}t'. \tag{6}$$

If the initial position $\vec{x}_0$ at time t_0 is known, then we can track the interface by simply integrating (5).

3. The discrete mixed markers and VOF method

3.1. The discrete convection equation for the color function C

Let Ω be now a bounded rectangular domain with $n_x \times n_y$ square cells with grid spacing $\Delta x = \Delta y = h$. The reference phase is contained in the region $\Omega_{1h} \subset \Omega$ whose boundary S_h consists of a continuous chain of segments. In this paper we consider a subdomain Ω_{1h} that is simply connected, but a generalization to

non-simply connected sets can be done in a straightforward manner. Let $\vec{u}_h$ be the discretized flow field over the mesh and we assume that the field $\vec{u}_h(\vec{x},t)$ can be constructed linearly from the values $\vec{u}_h$ on the cell vertices, or on the cell side midpoints of a staggered MAC grid, in a finite element fashion (see for example [1]). Then, without restrictions the discrete field $\vec{u}_h$ is continuous and weakly differentiable at all points. Let χ_h be the phase indicator function over the discrete domain Ω_{1h} and $C_{ij}(t)$ the discrete color function ($i=1,\ldots,n_x$, $j=1,\ldots,n_y$) defined by

$$C_{ij}(t)=\frac{1}{A_{ij}}\int_{A_{ij}}\chi_h(\vec{x},t)\,\mathrm{d}\vec{x}, \tag{7}$$

where A_{ij} is the area (the volume V in three dimensions) of cell (i,j). Clearly, the color function $C_{ij}(t)$ is one for all cells located in the interior of Ω_{1h}, zero for all external cells and between zero and one in the cells cut by the interface. The basic equation for algebraic and geometric VOF methods is the discretization of (3) or its integral form (4). In the geometric approach we need to find the discrete indicator χ_h which satisfies the following equation:

$$\frac{\partial}{\partial t}\int_{A_{ij}}\chi_h\,\mathrm{d}\vec{x}+\int_{S_{ij}}\vec{u}_h\cdot\vec{n}\chi_h\,\mathrm{d}\vec{s}=0\quad\forall A_{ij}, \tag{8}$$

or, with the introduction of $C_{ij}(t)$,

$$A_{ij}\frac{\partial C_{ij}(t)}{\partial t}+\int_{S_{ij}}\vec{u}_h\cdot\vec{n}\chi_h\,\mathrm{d}\vec{s}=0\quad\forall A_{ij}. \tag{9}$$

We remark that (9) cannot be written only as a function of $C_{ij}(t)$. In fact the color function does not contain the precise information on the interface location which are necessary to compute the reference phase fluxes across the cell boundary. Classical geometric VOF algorithms base their computations only on the color function C. The methods are implicit, since the field C is inverted to find an estimate $\widetilde{\chi}_h$ of χ_h which determines an approximate interface position. To describe briefly this procedure it is often somewhat incorrectly stated in the literature that the color function C obeys the same advection equation (3) which is only valid for χ, but that a reconstruction of the interface is also necessary to calculate its fluxes across the cell sides. For PLIC methods the reconstruction is basically a two-step procedure. In a given cell (i,j) the normal $\vec{n}$ is first determined from the knowledge of the color function in this cell and the neighboring ones. This step is not unique and extensive reviews of several methods for its calculation in two dimensions have been given in [17,24,28]. Then the segment approximating the interface is moved along the normal direction to enforce volume conservation [24,27], in other terms until the estimate $\widetilde{\chi}_h$ satisfies (7). The reconstructed piecewise-linear interface is discontinuous in general across the cell boundary and cannot reproduce interface details inside the cell. This is the case when the local radius of curvature is comparable to or smaller than the grid spacing h or when the interface is very convoluted, for example a filamentary structure.

3.2. The discrete equation of motion of the interface for marker methods

For marker methods the appropriate equation for interface tracking is the Lagrangian formulation of (6). In this case we are interested to determine the temporal evolution of the boundary $S_h(\vec{x},t)$ of the subdomain Ω_{1h}. Let $\{\vec{x}_k\in S_h,\ k=1,2,\ldots,N_{ij}\}$ be the marker set in the cell (i,j). In two-dimensional geometry we limit the number of markers N_{ij} for cell in the range from 4 to 6. These markers can be classified in two main categories: intersection markers and conservation markers. As we will explain later on in this section, intersection markers are the markers intersecting the grid lines, while conservation markers are

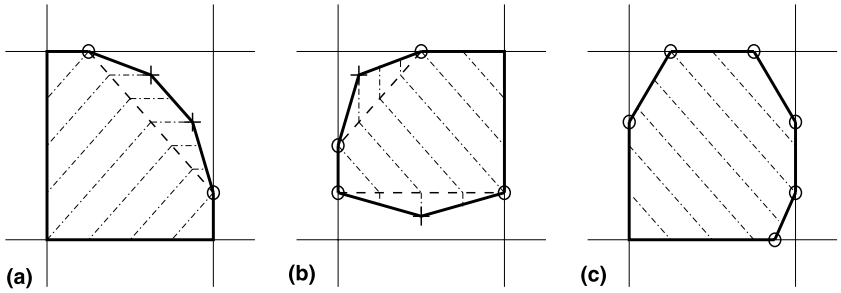

Fig. 1. Three different configurations for cells cut by the interface with intersection markers denoted by a circle and conservation markers by a cross: (a) standard configuration with two intersection and two conservation markers; (b) configuration with four intersection and two conservation markers; (c) case with six intersection markers and no conservation marker.

added inside the computational cell to conserve the volume fraction. Let $N_{\mathrm{I},ij}$ and $N_{\mathrm{C},ij}$ be the number of intersection and conservation markers in the cell (i,j), then $N_{\mathrm{I},ij} + N_{\mathrm{C},ij} = N_{ij} \leqslant 6$, with $N_{\mathrm{I},ij} \geqslant 2$. In Fig. 1 three possible cell configurations are represented, with the intersection markers denoted by a circle and the conservation markers by a cross. The standard configuration in Fig. 1(a) has two intersection markers and two conservation markers, while the two more complex configurations of Figs. 1(b) and (c) have four intersection and two conservation markers and six intersection markers, respectively.

Given the flow field $\vec{u}(\vec{x},t)$ if $\vec{x}_k$ is a marker of the interface inside the cell (i,j) we have

$$\frac{\mathrm{d}\vec{x}_k}{\mathrm{d}t} = \vec{u}(\vec{x}_k(t),t), \quad k = 1,2,\ldots,N_{ij}, \tag{10}$$

for all $i = 1,\ldots,n_x$ and $j = 1,\ldots,n_y$. Given the marker position at the initial time, we can track the interface point $\vec{x}_k$ by integrating (10) with a standard numerical method. Here the interface is supposed to be continuous and its geometrical properties can be computed for example by interpolating the surface points with continuous and differentiable functions.

3.3. Numerical algorithms

We now turn our attention to the numerical algorithms for the solution of (9) and (10). The algorithm appeals mainly to geometry, because the volume fluxes flowing through the cell sides over one time step are polygons formed by interface line segments and cell sides. These fluxes are computed in a straightforward and systematic manner by using standard formulas for straight lines intersecting polygons. The algorithm consists mainly of two parts: interface reconstruction and advection.

3.3.1. Interface reconstruction

The starting point for the reconstruction algorithm are the intersection markers on the grid lines and the updated color function C computed by the advection algorithm. At present, a cell cut by the interface is forced to have a minimum of two or a maximum of six intersection markers. The reconstruction inside a cut cell is then completed by adding the conservation markers. The procedure can be performed in different ways and our choice may not be optimal, but it is very simple. With reference to Fig. 2, we consider the standard situation where the interface has only two intersections on different sides of the cell (here denoted by A and B) which are also intersection markers. Two conservation markers (points c and d) are placed on the segment connecting A and B, each of them at 1/4 of the segment length from the closest end point, and then they are displaced along the normal direction to their final position (points C and D). The total area $S_1 + S_2$ equals the corresponding value $C_{ij}A_{ij}$ obtained from the advection algo-

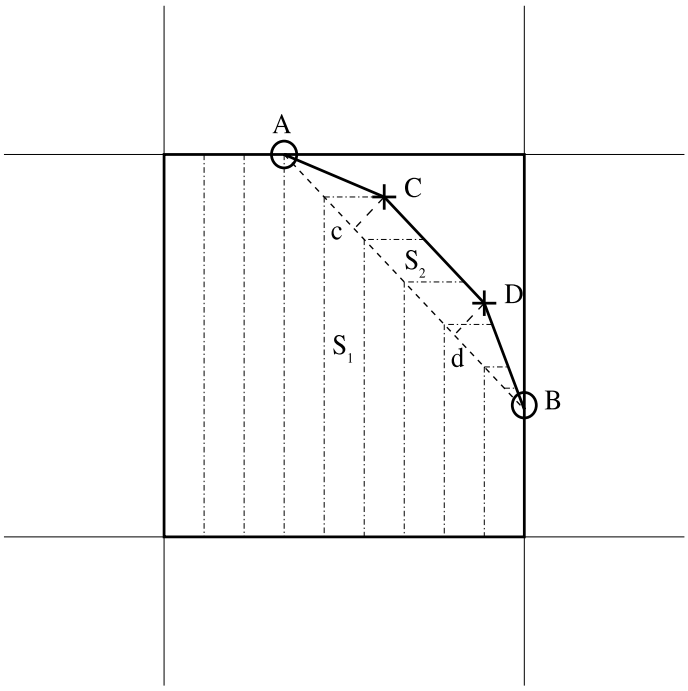

Fig. 2. Typical interface reconstruction in a cell with two intersection markers, denoted by a circle, and two conservation markers, denoted by a cross.

rithm of the color function and the constraint $cC = dD$ is satisfied. Moreover, $AC = DB \approx CD$ by construction, and in general the missing area S_2 is much smaller than S_1, so that the four markers A, C, D, B within a cell are almost evenly spaced along the interface. The resulting interface consists of a continuous chain of segments connecting the two types of markers and the ordering along the line is in the counterclockwise direction with the reference phase on the left. If the interface intersects four points on the cell boundary we add only one marker in the middle of each segment connecting a couple of consecutive points and the missing area S_2 is equally redistributed. One of these configurations is shown in Fig. 1(b), where the four intersection markers together with the two conservation markers and one cell vertex define a polygon containing the reference phase in its interior (hatched area of Fig. 1(b)). If the interface intersects more than four points in the same cell, no conservation marker is added and no volume conservation algorithm is applied. However, the occurring of such an event is considered rare and negligible for the total mass conservation. The reconstructed interface consists then of an ordered list of points, the intersection and conservation markers, that once connected by straight lines conserve the volume with a precision typical of VOF methods.

3.3.2. Interface advection and volume fraction update

In the advection step we update the color function C and calculate the new intersection markers. In each cell cut by the interface we determine a polygon containing the reference phase in its interior and then advect the four vertices of the cell in a Lagrangian way along the streamlines. To better illustrate the procedure, let us consider the cell (i, j) of Fig. 3 defined by the four vertices $\alpha, 2, 3, 4$. The reference phase is contained inside the polygon delimited by the following set of seven vertices: the two intersection markers 1,5, the two conservation markers 6,7 and the three cell vertices 2,3,4. The vertices are collected in counterclockwise order with the reference phase on the left and their coordinates are stored in normalized form,

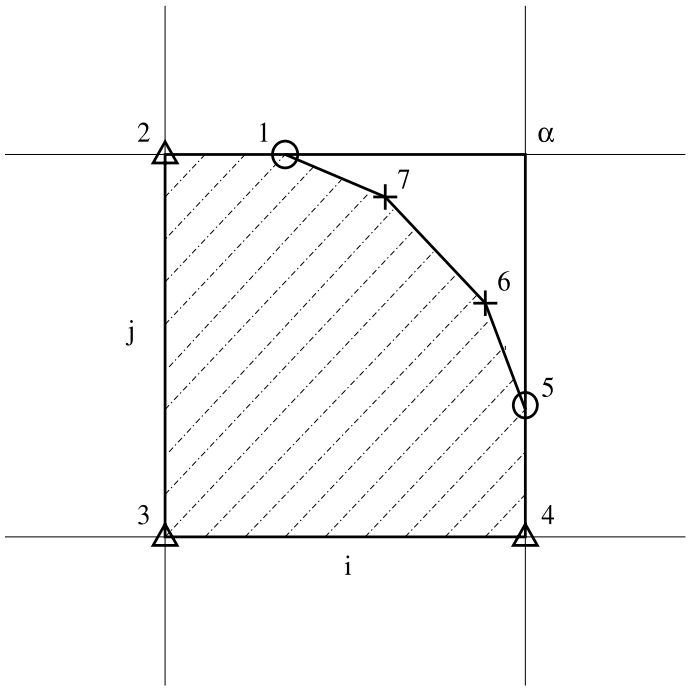

Fig. 3. The seven vertices denoted with the digits 1–7 define a polygon containing the reference phase in its interior. The vertices are the intersection markers (circles), the conservation markers (crosses) and three cell vertices (triangles).

for example the abscissa x_7 of the polygon vertex 7 can be determined from its normalized coordinates (ξ_7, η_7), with both ξ_7 and η_7 in the range $[0, 1]$, with the bilinear formula

$$x_7 = x_\alpha \xi_7 \eta_7 + x_4 \xi_7 (1 - \eta_7) + x_2 (1 - \xi_7) \eta_7 + x_3 (1 - \xi_7)(1 - \eta_7). \tag{11}$$

Then, as shown in Fig. 4, the four vertices of the cell (i, j) are advected to the points $\alpha', 2', 3', 4'$ along the streamlines with a numerical integration of (10). We use a standard fourth-order Runge–Kutta method (described in [19]) and the local velocity in a point (ξ, η) inside the cell is obtained with a bilinear interpolation of the components of the velocity field which can be defined either in the center of the cell sides, as in a staggered MAC grid, or on the cell vertices. The new coordinates of the polygon vertices are calculated with expression (11) with the same normalized coordinates, but with the updated values of the coordinates of the cell vertices. The polygon vertices are again connected with straight lines and the intersections a, b, c, d with the grid lines are then determined. The reference phase originally inside the cell (i, j) after advection is displaced over several neighboring cells. With reference to Fig. 4, we see that the contribution from cell (i, j) at time n to the total area $C_{i,j+1} A_{i,j+1}$ of cell $(i, j + 1)$ at the following time $n + 1$ is the area A_p of the polygon defined by the five vertices $1', 2', a, \alpha, d$ which is given by the expression [24]

$$A_p = \frac{1}{2} \sum_{k=1}^{N} (x_k y_{k+1} - x_{k+1} y_k), \tag{12}$$

where N is the number of vertices of the polygon and the vertex $N + 1$ coincides with the first one. Similarly, we calculate the other three hatched areas in Fig. 4 which belong to different grid cells. The

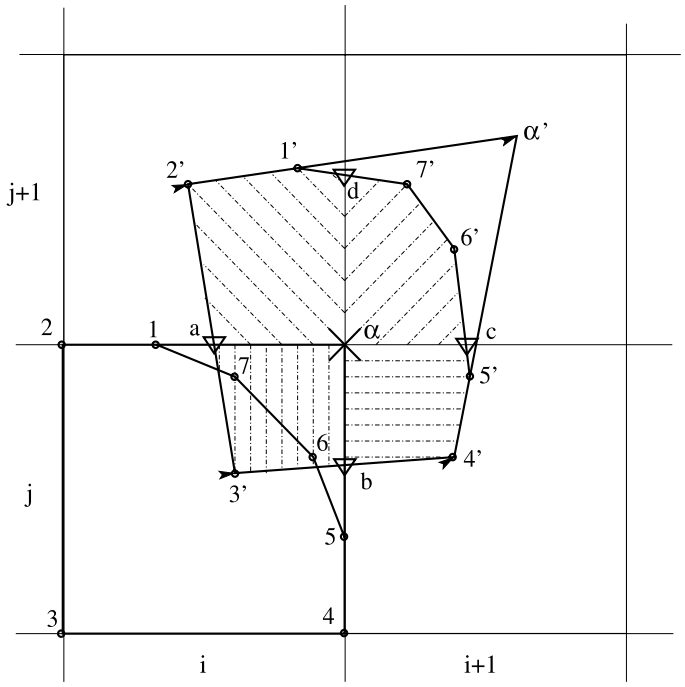

Fig. 4. The four vertices of the cell (i, j) are advected along the streamlines to points $2', 3', 4', \alpha'$. The seven vertices $1'$–$7'$ of the updated polygon, which contains the reference phase, are calculated with a bilinear expression of the coordinates of the advected cell vertices. The four points a, b, c, d are the intersections of the advected polygon with the grid lines. The area of the hatched regions represents the contribution from cell (i, j) at time n to the reference phase in the four neighboring cells at time $n + 1$.

process is repeated over all cells to get the updated color function C. The new intersection markers are calculated during the advection step as represented in Figs. 4 and 5. Only the two intersections c, d of the advected polygon with the grid lines are intersection markers, since they are part of the interface, this is not the case for the other two intersections a, b which lie, before the advection, on the boundary of cell (i, j). With the updated color function C and the new intersection points the interface reconstruction is completed with the addition of the conservation markers, as described in the previous section. As an example, we can consider the cell $(i + 1, j + 1)$ in Fig. 5. The two intersection markers c, d play the role of points A, B of Fig. 2, points $6', 7'$ are discarded in the interface reconstruction, and two new conservation markers are added on the segment cd and displaced in the normal direction until the total area occupied by the reference phase is equal to the value $C_{i+1,j+1}A_{i+1,j+1}$. If we do not add any conservation marker, in order to conserve the area we are obliged to move the whole segment cd along the normal direction, recovering in this way a PLIC reconstruction that will not be continuous across the cell boundary. On the other hand, the more points we add on the interface, the less we need to move them around inside a cell to conserve the area. We have found that the choice of two conservation markers is a good compromise between the computational efficiency of the method and its accuracy in the interface description.

So far we have described the reconstruction and advection of an interface with a local radius of curvature much bigger than the grid size so that the interface has only two intersections with the cell boundary. When the local radius of curvature becomes smaller than h or when the interface line is locally compressed and many markers cluster together, it may occur that there are more than two intersections on

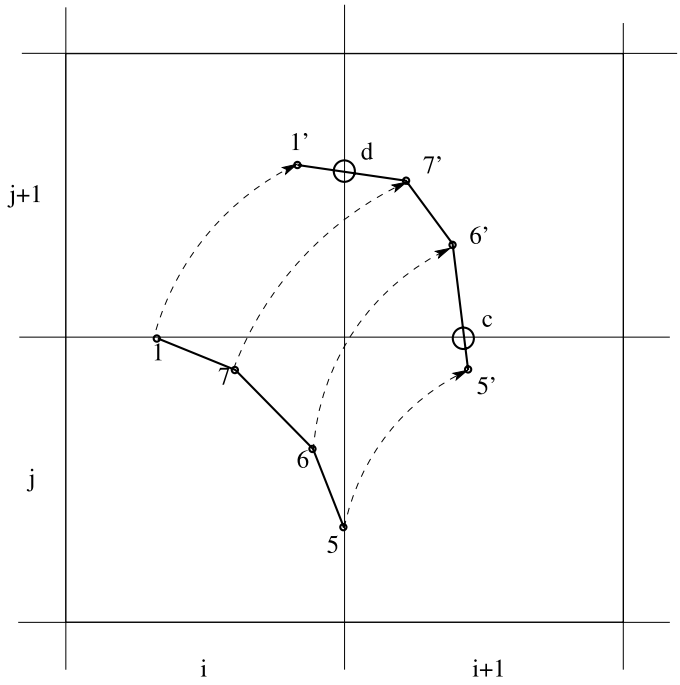

Fig. 5. The intersection and conservation markers 5,6,7,1 after the advection step move to points $5', 6', 7', 1'$ and define the two new intersection markers c, d, denoted by a circle.

the same cell side and the interface shape is not well resolved. To reduce the complexity of the algorithm we limit the number of intersection markers to a maximum of six and no more than two of them on the same cell side. The markers reduction algorithm, which operates in very particular and rare situations, that is currently implemented in our method is depicted in Fig. 6, where we show two typical instances in which the number of intersection markers must be reduced. On the left the three intersections on the same side are replaced by a single marker in the middle, while on the right the four intersections are reduced to only two markers.

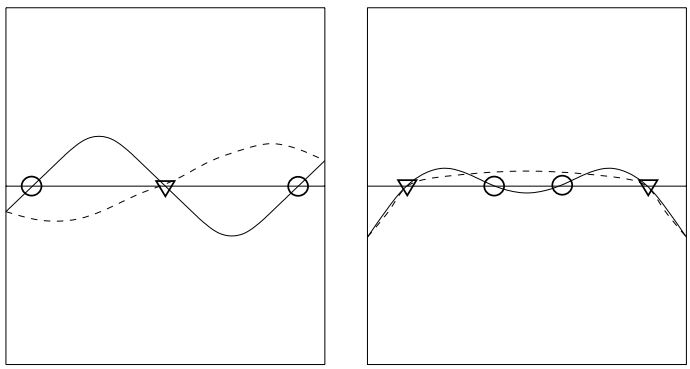

Fig. 6. Two examples of the markers reduction strategy: on the left, three intersection points on the same cell side (the two circles and the triangle) are reduced to a single one in the middle (triangle), while on the right four points (circles and triangles) are reduced to only two (triangles).

We conclude the description of our hybrid method with a qualitative comparison with VOF techniques in terms of number of operations necessary to advect the interface. In each cell we advect the four vertices with a fourth-order Runge–Kutta method, which is by far the most expensive part of the algorithm. As explained in the next section, we divide the time step in several smaller steps for a more accurate integration along the streamlines. Each single integration requires the evaluation of the velocity in four different locations with a bilinear formula similar to (11). The new coordinates of the markers inside the advected cell are calculated with a bilinear expression as well. The intersections of the polygon with the grid lines and the position of the new conservation markers are simple and straightforward calculations. The area contribution from the advected polygon to a neighboring cell requires the gathering of the vertices in a counterclockwise order and the evaluation of expression (12). The first step in a PLIC method is the reconstruction of the interface, written as $\vec{n} \cdot \vec{x} = \alpha$ in each cut cell. The interface normal $\vec{n}$ is a straightforward calculation for first-order accurate schemes, but it can be rather lengthy for iterative second-order accurate reconstruction algorithms. The value of the line constant α comes from enforcing volume conservation and in two-dimensional Cartesian geometry is a relatively simple task [27]. The interface reconstruction is required in several advection algorithms, either one-dimensional or multidimensional, to define the total area crossing each cell side and to determine the portion of this area occupied by the reference phase. The updated value of the color function C is obtained with a simple balance between incoming and outgoing fluxes. If we limit the number of subdivisions of the time step to one or two, then we find that the run time of our hybrid scheme is comparable to that of a second-order accurate VOF algorithm.

4. Numerical tests

In this section we analyze the performance of our mixed markers and volume tracking algorithm over standard two-dimensional tests. Literature surveys indicate that uniform translations and rotations are the simplest flow fields that are widely used to test an interface tracking algorithm. Typical examples of such tests can be found in [10,14,17,24,25,28,40]. An acceptable tracking method should translate and rotate smooth fluid bodies without significant distortion or deterioration of the fluid interface. Also, mass should be conserved rigorously. More challenges arise when we advect in these velocity fields geometrical figures such as squares, crosses and slotted disks, since VOF methods tend to smooth corners into rounded shapes. In all cases, it is usually tested the convergence to the real shape with grid refinement. Flows inducing simple translations or rotations of fluid bodies with no distortion do not adequately challenge interface tracking methods which are designed for topology changes. Translations and rotations should be considered as useful debugging tests, but they are not sufficient for a final judgment of the tracking algorithm. In a real dynamical situation, sharp gradients in fluid properties lead to interfacial unstable flows, such as the Rayleigh–Taylor and Kelvin–Helmholtz instabilities, which are characterized by a not uniform vorticity field. A number of tests have been proposed by several authors [3,23,24,31] and we consider two of them. These two-dimensional problems impose strong and not uniform vorticity fields, that induce gross distortions in the interface shape eventually leading to topology changes, to challenge the algorithm capabilities and for a proper and quantitative assessment of their performances. The first test contains a single vortex in a unit box that deforms progressively an initial circular shape spinning and stretching it into a filament that spirals toward the vortex center. In the second test problem we consider the flow field characterized by either 4×4 or 8×8 vortices. The interface line undergoes a strong distortion with the development of several filaments. However, in the analytical solution and in the converged limit of the simulations, these filaments do not tear. In this way, it is tested the capability of the algorithm to represent thin structures without artificial breakup or coalescence. For the purpose of error analysis, in the tests with not uniform vorticity we use a cosinusoidal time-dependence so that the

fluid returns to its initial state after half a period [15]. To compare our results with those in the literature we introduce two widely used L_1 error norms. The first one is the relative mass error $E_m(t_1)$ between the total volume of the reference phase at the initial time t_0 of the simulation and that of the deformed fluid body at time t_1 defined as

$$E_m(t_1) = \frac{|\sum_{ij} A_{ij} C_{ij}(t_0) - \sum_{ij} A_{ij} C_{ij}(t_1)|}{\sum_{ij} A_{ij} C_{ij}(t_0)}, \tag{13}$$

the second one is the geometrical error $E_g(t_1)$ between the position of the fluid body at the initial time t_0 and at time t_1 defined as

$$E_g(t_1) = \sum_{ij} A_{ij} |C_{ij}(t_0) - C_{ij}(t_1)|. \tag{14}$$

Our computational cells are all equal and square, so in the previous two expressions we can substitute A_{ij} with h^2. Moreover, the geometric error has units of area in two dimensions and from its change with grid refinement we can infer rates of convergence [24]. The geometric error is particularly relevant to the tests where the final shape of the fluid body should be equal to the starting one.

4.1. Translations and solid body rotations

Translational and rotational flows do not induce interface distortion and the volume fraction field evolution associated to these velocity fields can be calculated exactly. For both flows, we place a circular shape in the unit box that is partitioned with $n_x \times n_y$ equal, square cells. In particular, we have $n_x = n_y$, $h = 1/n_x$ and $n_x = 16, 32, 64, 128$. The volume fraction is set to one in the cells inside the circle, zero outside and it is calculated analytically in the cut cells. The intersection markers are initially determined with a least-square technique that interpolates the midpoints of the segments of a PLIC reconstruction with a parabolic function. Conservation markers are then added to obtain the correct C value. For all tests, we use a fourth-order Runge–Kutta time-integration scheme over the time $\delta t = CFL/4$. The CFL number is the non-dimensional velocity $CFL = u\Delta t/h$, where u is the velocity component along the x axis, Δt the time step and h the grid spacing. For example, if $CFL = 1$, the circle is displaced one grid spacing per time step along the x direction. This means that we apply the Runge–Kutta scheme four times for each time step Δt. Experimentally, this appears to be a good compromise between computational efficiency and accuracy of both mass and geometric errors that decrease as the markers are advected more accurately along the streamlines. In real dynamical simulations $CFL < 1/4$ and the numerical integration scheme can be applied only once. Furthermore, it is also simple to adapt the number of integrations to the local CFL number.

4.1.1. Translations

A uniform and constant in time velocity field (u, v) is imposed with opposite components, $u = -v$, so that both phases translate diagonally across the mesh. The reference phase is inside a circle of radius $r = 0.15$ that is initially centered at $(0.25, 0.75)$. It returns to its initial position after one domain translation allowing mass and geometric error measurements with (13) and (14), respectively. The circle should not change its shape as a result of this motion. In Fig. 7 we show its initial shape and after half translation when the velocity field is instantaneously inverted to bring back the circle to its initial position and complete one domain translation. In Table 1 we compare the geometrical error E_g of our hybrid algorithm with that obtained in [24], where the authors use a VOF algorithm based on a least-square method to calculate the normal vector $\vec{n}$ and an unsplit time-integration scheme to advect the interface. The results are for two domain translations (actually, our results compare even more favorably, since in [24] the circle has radius 0.25, so we should consider an intermediate error between those obtained with $n_x = 32$ and $n_x = 64$).

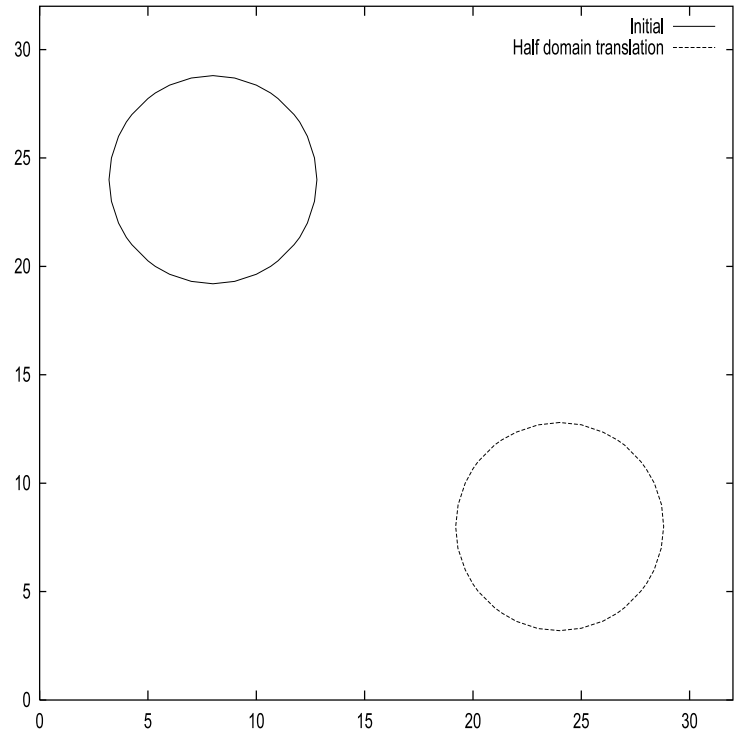

Fig. 7. Initial circular shape (solid line) and after half diagonal translation (dashed line).

In the same table we show the mass error E_m and the geometrical error E_g after 10 translations. We remark that the mass error is below machine error and our geometric error is consistently better than that obtained with standard VOF algorithms. A good discussion on VOF algorithm errors and their convergence rates

Table 1
Mass (E_m) and geometric (E_g) errors for two complete translations along the main diagonal at different resolutions and *CFL* numbers

n_x	*CFL*	E_m	E_g	E_g (RK)	E_g (10)
32	1.0	0.	0.		0.
	0.5	0.	9.51e − 5	6.21e − 4	3.14e − 4
	0.1	0.	1.30e − 4		3.19e − 4
	0.01	0.	1.08e − 4		3.55e − 4
	0.001	0.	1.29e − 4		3.83e − 4
64	1.0	0.	0.		0.
	0.5	0.	3.35e − 5	2.47e − 4	9.97e − 5
	0.1	0.	3.97e − 5		8.56e − 5
	0.01	0.	3.13e − 5		8.93e − 5
128	1.0	0.	0.		0.
	0.5	0.	1.58e − 5	5.10e − 5	6.44e − 5
	0.1	0.	1.34e − 5		4.40e − 5
	0.01	0.	7.57e − 6		1.90e − 5

The data in the fifth column are taken from [24]. In the last column the geometric error for 10 translations.

can be found in [10]. In particular, a continuous chain of segments with at least three of them in each computational cell cut by the interface is much better than a standard PLIC reconstruction, which is piecewise-linear with just one segment in a cell, but still it does not reproduce exactly a curved line. However, here we are mainly interested in the dependence of the geometric error with the *CFL* number. Results for several *CFL* numbers at different grid resolutions are again given in Table 1. In standard VOF methods the reconstruction error is minimum at $CFL = 1$ and progressively increases as the *CFL* number is decreased, but it remains bounded as the *CFL* number approaches zero [10]. This asymptotic behavior is essential for the viability of VOF methods, since in real dynamical simulations with explicit time-integration techniques the *CFL* number is usually much smaller than one, for numerical stability conditions. The asymptotic value changes as a function of the reconstruction technique and grid refinement. It was also pointed out that when a piecewise-linear interface is advected, most of the error in the scalar field C is generated from the regions near the interface discontinuity at the cell boundary [28]. Moreover, a PLIC reconstruction is based only on the C data at a given time and it has no memory of the previous reconstructions. This is not the case with a marker approach, but we introduce an error when we remesh the markers, that we call displacement error. Since the integration along the streamlines is exact for a simple translation, this error remains constant during a single time step integration. As the markers are displaced from their original position to generate new intersection and conservation markers, a discrete amount of error is added. The displacement error is zero at $CFL = 1$, since there is no remeshing, and we expect it asymptotically to decrease with grid refinement and with the time step, since in these conditions the displacement approaches zero. This analysis is supported by the results in Table 1. At low resolution, $n_x = 32$, and for 10 translations the geometric error increases while lowering the *CFL* number, but still it does not

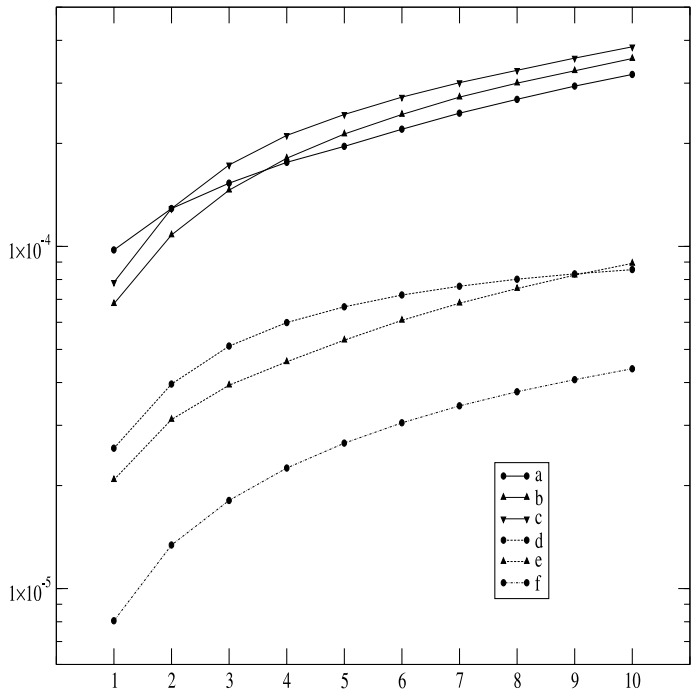

Fig. 8. Geometric errors on a 32^2 grid (solid lines) at $CFL = 0.1$ (a), $CFL = 0.01$ (b) and $CFL = 0.001$ (c), on a 64^2 grid (dashed lines) at $CFL = 0.1$ (d) and $CFL = 0.01$ (e) and on a 128^2 grid (dotted line) at $CFL = 0.1$ (f), as a function of domain translations.

seem to have reached an asymptotic value. At high resolution, $n_x = 128$, the displacement error is in the asymptotic region and the geometric error diminishes with the *CFL* number. In the other intermediate cases the geometric error is fluctuating, without showing a definite behavior with the *CFL* number. This picture is further supported by the results of Fig. 8 where the geometric error has been plotted as a function of the number of diagonal translations. The error line at a lower *CFL* crosses the line at a higher *CFL* number in a point. Starting from this crossing point the geometric error increases while lowering the *CFL* number, as in a standard VOF method, because of the greater number of reconstructions and marker remeshings. However, as we increase the grid resolution, a bigger number of translations is necessary to reach the crossing point between two lines at different *CFL*.

4.1.2. Solid body rotations

4.1.2.1. Case A: rotation of a circle. A constant-vorticity velocity field is imposed with the rotation center at $(0.5, 0.5)$

$$\begin{aligned} u &= \Omega(0.5 - y), \\ v &= \Omega(x - 0.5). \end{aligned} \tag{15}$$

This solenoidal field rotates the fluid body in the counterclockwise direction around the domain center. The initial shape, a circle centered at $(0.50, 0.75)$ with a radius $r = 0.15$ as in the translation test, is plotted in Fig. 9 and after half rotation. Afterwards, the fluid returns to its initial configuration, allowing error measurements with (13) and (14). The circle center describes in one rotation a path of about the same length of the diagonal of the computational unit box. The results for the rotation of a circular body are presented in Table 2, where the *CFL* number is based on the maximum velocity in the computational domain. The

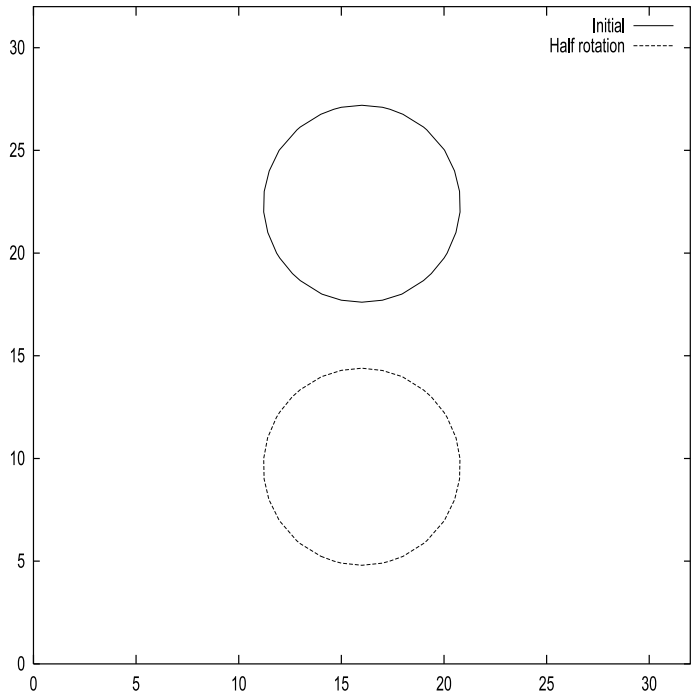

Fig. 9. Initial circular shape (solid line) and after half rotation (dashed line).

Table 2
Mass (E_m) and geometric (E_g) errors for one rotation at different resolutions and *CFL* numbers

n_x	*CFL*	E_m	E_g	E_g (RK)	E_g (10)
32	1.0	0.	1.23e − 4		9.60e − 4
	0.5	0.	1.31e − 4	1.61e − 3	9.80e − 4
	0.1	0.	1.45e − 4		1.37e − 3
64	1.0	0.	2.89e − 5		2.65e − 4
	0.5	0.	3.23e − 5	3.54e − 4	2.90e − 4
	0.1	0.	3.68e − 5		3.55e − 4
128	1.0	0.	7.40e − 6		6.65e − 5
	0.5	0.	8.90e − 6	8.95e − 5	8.43e − 5
	0.1	0.	1.13e − 5		1.07e − 4

The data in the fifth column are taken from [24]. In the last column the geometric error for 10 rotations.

streamlines are now curved but the fluid body is not deformed, and for such a simple flow field our tracking algorithm conserves the volume to machine error. The geometrical error after one revolution is shown in the fourth column and it has about the same value of the translation test, but for a twice as long distance. It is also about one order of magnitude smaller than in [24], that is comparable with the geometric error we obtain with 10 revolutions. From the value of E_g after 10 rotations, shown in the last column, we notice that the geometrical error grows faster than in the translation case. Moreover, in the rotation test E_g is different

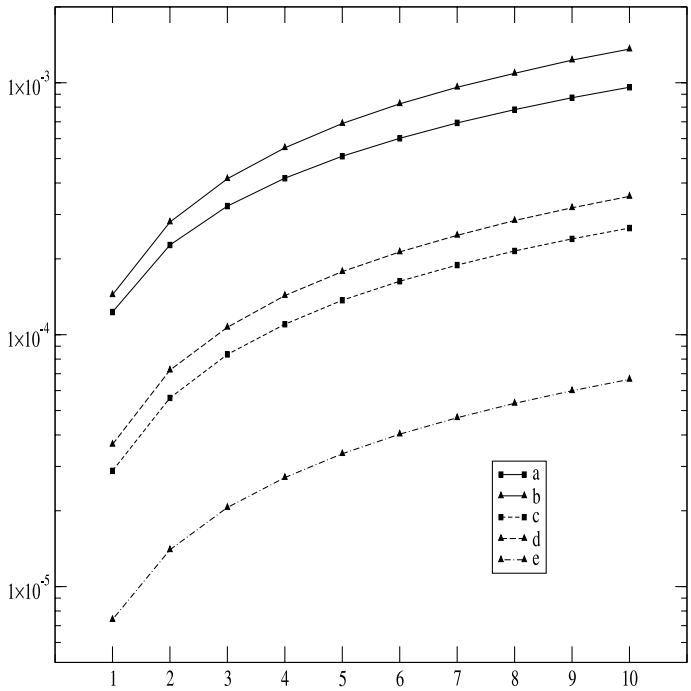

Fig. 10. Geometric errors on a 32^2 grid (solid lines) at $CFL = 0.1$ (a) and $CFL = 0.01$ (b), on a 64^2 grid (dashed lines) at $CFL = 0.1$ (c), $CFL = 0.01$ (d) and on a 128^2 grid (dotted line) at $CFL = 0.1$ (e), as a function of domain rotations.

from zero even at $CFL = 1$. In fact, the streamlines are now curved and the marker remeshing is necessary, whatever the value of the *CFL* number. This results in a bigger geometrical error, which reaches now an asymptotic value as shown in Table 2 and that increases while lowering the *CFL* number as in a standard VOF method. This is also evident from the lines in Fig. 10 which represent the variation of the geometric error with the number of revolutions. The lines never cross, and those with a higher *CFL* systematically have a smaller geometric error.

4.1.2.2. Case B: rotation of a square. We now consider the solid body rotation of a square and check the ability of our method to cope with corners. We divide the unit box in 100^2 cells as in the Zalesak slotted disk test, the square center is at $(0.5, 0.75)$ and the side width is 21 cells. The velocity field is again given by (16) and a full rotation is now performed in 400 steps. Standard VOF methods progressively smooth the corners into a rounded shape as the simulation goes on. The initial shape of the square is depicted in Fig. 11 where it is also compared with our solution after 1, 5 and 10 rotations, respectively. The corners are rounded in

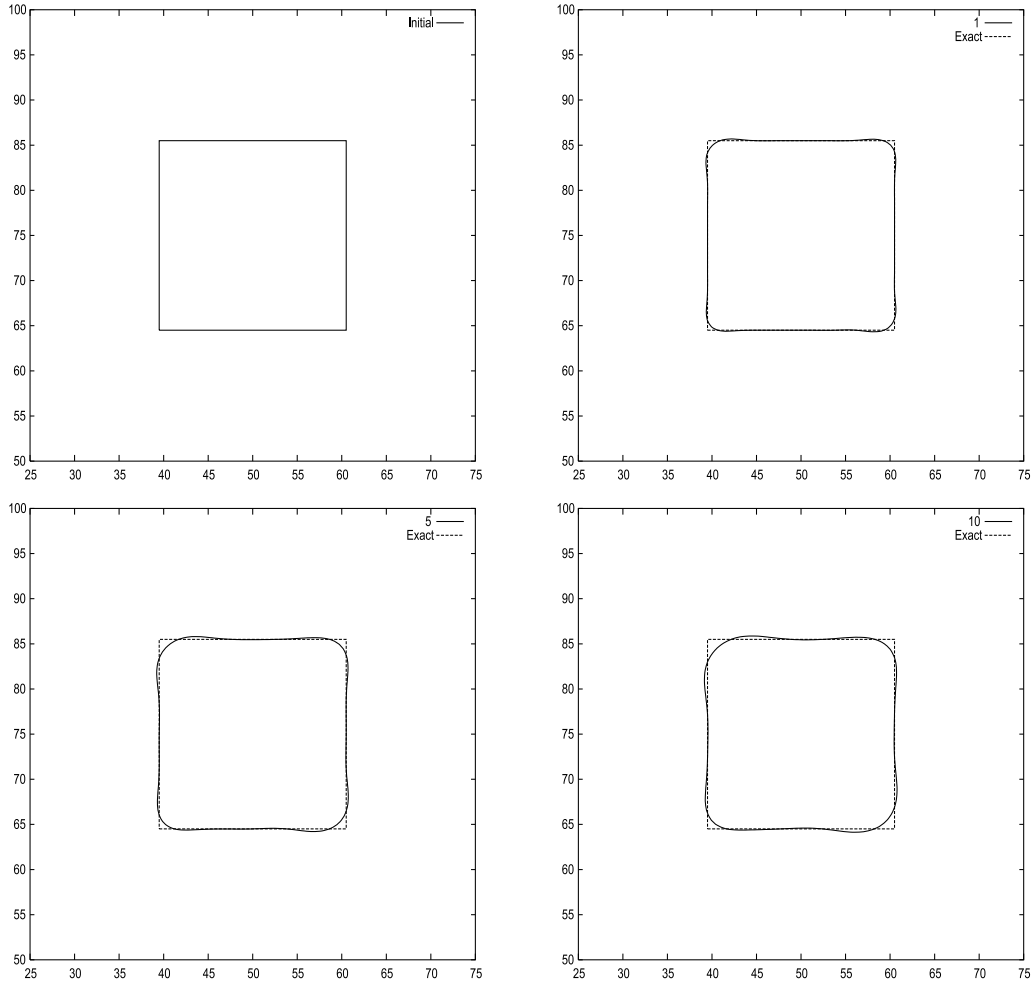

Fig. 11. Square rotation test. Initial square shape (top left) and after 1 (top right), 5 (bottom left) and 10 (bottom right) rotations.

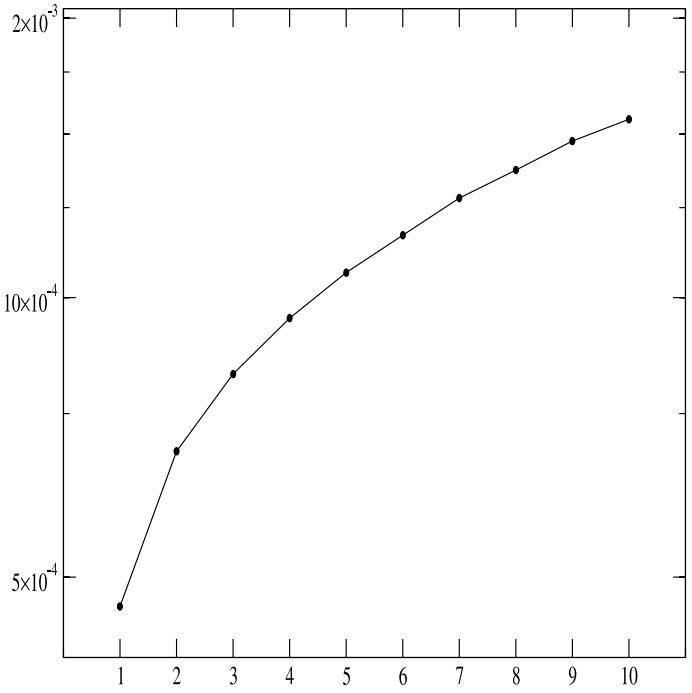

Fig. 12. Geometrical error for the square on a 100^2 mesh at $CFL = 0.1$, as a function of the number of rotations.

our hybrid technique as well, but the comparison is extremely favorable against VOF methods. A system of fixed markers at the corners would be necessary to keep the sharp corners after each reconstruction. However, in a real dynamical simulation capillary effects due to surface tension will quickly remove regions with very high curvature that may exist during the breakup or coalescence of bubbles and droplets. In Fig. 12 we show the evolution of the geometrical error as a function of the number of rotations at $CFL = 0.1$.

4.2. Single vortex test

Simulations of translation and rotations do not test the ability of the method to accurately resolve thin filaments, which may develop in stretching and tearing flows, when their thickness is comparable to the grid spacing h. The single vortex test or "vortex-in-a-box" problem has been widely used in the literature (for example in [3,6,10,24,28]) to check the ability of the tracking algorithm to treat interfaces which undergo strong deformations. The reference phase is inside a circle with center at $(0.5, 0.75)$ and radius $r = 0.15$. The velocity field in the unit domain is defined by the stream function

$$\psi = \frac{1}{\pi}\sin^2(\pi x)\sin^2(\pi y)\cos\left(\frac{\pi t}{T}\right). \tag{16}$$

The single vortex velocity field is centered in the box with the largest velocity located half way from the box center to the walls of the square domain. The cosinusoidal time-dependance has been introduced following [15]. Analytically, the interface is stretched up to time $t = T/2$, when the deformation is at a maximum, and then is brought back to its initial configuration at time $t = T$. Therefore, an indication

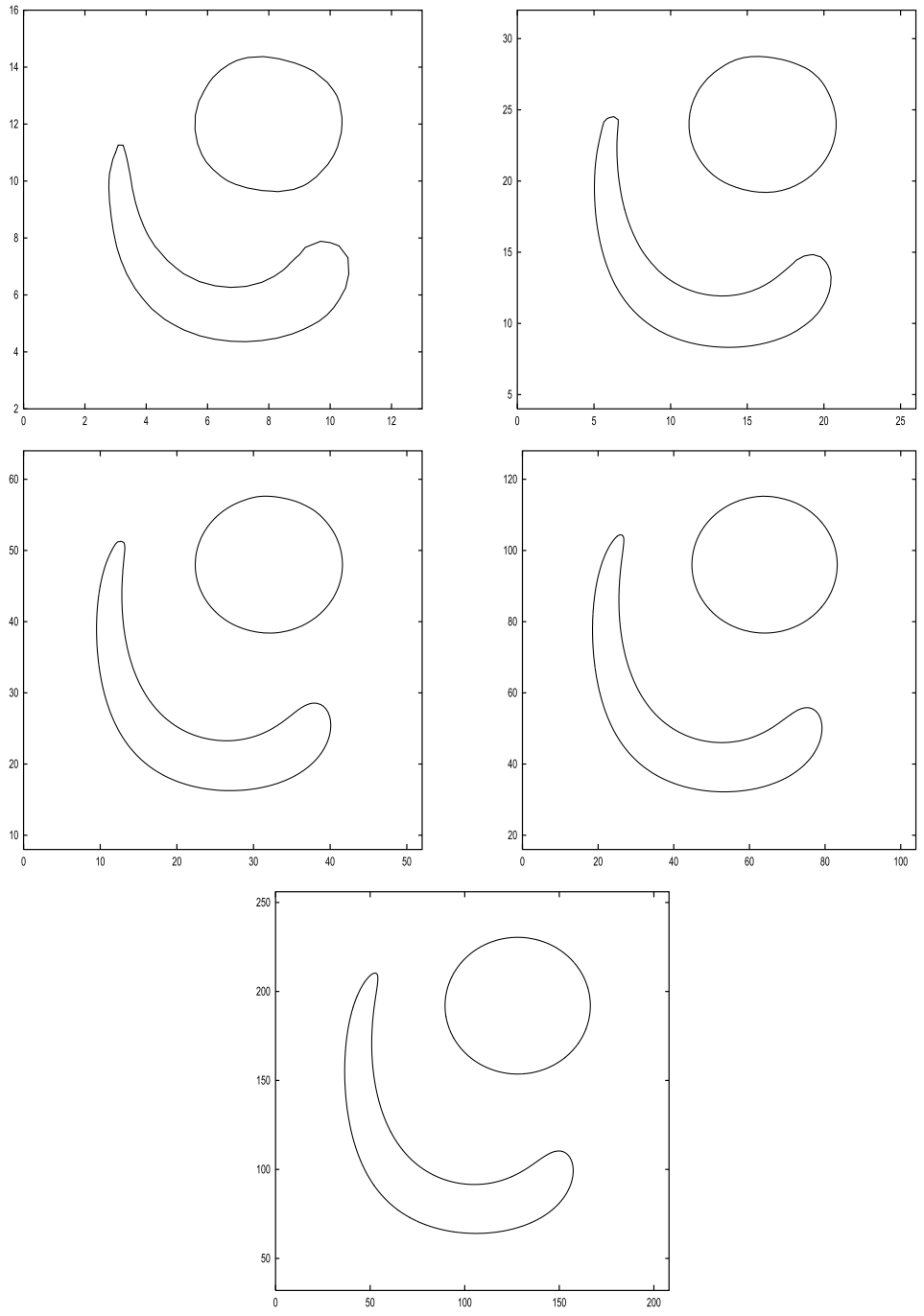

Fig. 13. The reconstructed interface at maximum deformation at $t = 1.0$ and back to the initial position at $t = 2.0$ for the single vortex field test with $T = 2$, for the following meshes with 16^2 (top left), 32^2 (top right), 64^2 (middle left), 128^2 (middle right) and 256^2 (bottom) cells.

Table 3
Mass (E_m) and geometric (E_g) errors for the single vortex test with $T = 2$, at different resolutions and *CFL* numbers

n_x	*CFL*	E_m	E_g	E_g (RK)	E_g (HF)
16	1.0	4.26e − 2	3.79e − 3		
	0.1	4.91e − 3	1.60e − 3		
	0.01	5.00e − 4	1.56e − 3		
	0.001	5.84e − 5	1.60e − 3		
32	1.0	7.96e − 3	1.00e − 3	2.36e − 3	2.37e − 4
	0.1	9.79e − 4	6.38e − 4		
	0.01	1.07e − 4	5.63e − 4		
	0.001	2.02e − 5	5.68e − 4		
64	1.0	1.59e − 3	2.69e − 4	5.85e − 4	5.65e − 4
	0.1	2.02e − 4	1.57e − 4		
	0.01	1.97e − 5	1.28e − 4		
128	1.0	2.95e − 4	5.47e − 5	1.31e − 4	1.32e − 4
	0.1	5.08e − 5	1.61e − 5		
	0.01	4.99e − 6	1.09e − 5		
256	1.0	6.99e − 5	1.36e − 5		
	0.1	1.22e − 5	3.27e − 6		

The data in the last two columns are taken from [24] and [10], respectively.

of the accuracy of the algorithm in terms of mass and geometric errors can be obtained by considering (13) and (14) between the initial and final volume fraction distribution. The most common reversal times T used in the literature are $T = 2$ and $T = 8$. We show in Fig. 13 the interface reconstruction at $t = 1.0$ and $t = 2.0$, when $T = 2$ for different grid resolutions, in particular on a 16^2, 32^2, 64^2, 128^2 and 256^2 mesh, respectively. Note that in all cases the solution is quite good. With this figure size, the finite grid resolution can only be appreciated near the tail and the head of the interface in the smallest two meshes, but the final interface reconstruction at $t = T = 2$ shows a stable and correct pattern even in the 16^2 mesh. In Table 3 we present our results for $T = 2$ and those obtained in [10,24] with two VOF/PLIC methods which were based on a similar reconstruction technique [17], but with two completely different unsplit advection algorithms. We observe that the mass conservation error with our algorithm decreases considerably with the *CFL* number and grid resolution. This is because the streamlines are better resolved at low *CFL* by the fourth-order Runge–Kutta algorithm and, moreover, the discrete interpolated velocity field is closer to be divergence-free with grid refinement. The geometric error shows a similar behavior. Initially, it decreases with the *CFL* number and grid refinement, because the trajectories are better resolved and the displacement errors for such a demanding test are positively affected by lowering the *CFL* number and by increasing the mesh size, since in both cases the interface slows down in its deformation per time step. As the *CFL* is further decreased, E_g reaches a saturation regime, this is clearly seen in the two runs at the lowest resolution, $n_x = 16, 32$, but this is not yet the case for $n_x \geqslant 64$. Again from Table 3, we see that the mixed method compares favorably with the results obtained for the same test with two VOF/PLIC unsplit algorithms [10,24]. Our hybrid method is capable to follow with great detail the interface motion, as in typical marker methods, and at the same time has a good mass conservation, as in a standard VOF/PLIC method. The results for $T = 8$ are shown in Fig. 14 just before the velocity reversal and at the end of the simulation, and the mass and geometrical errors are given in Table 4. At low resolution, the tail and head of the interface are not well resolved, they develop a bulge and lag the actual solution. This is because these regions have the highest local curvature and a few cells have more than six intersections. The marker reduction

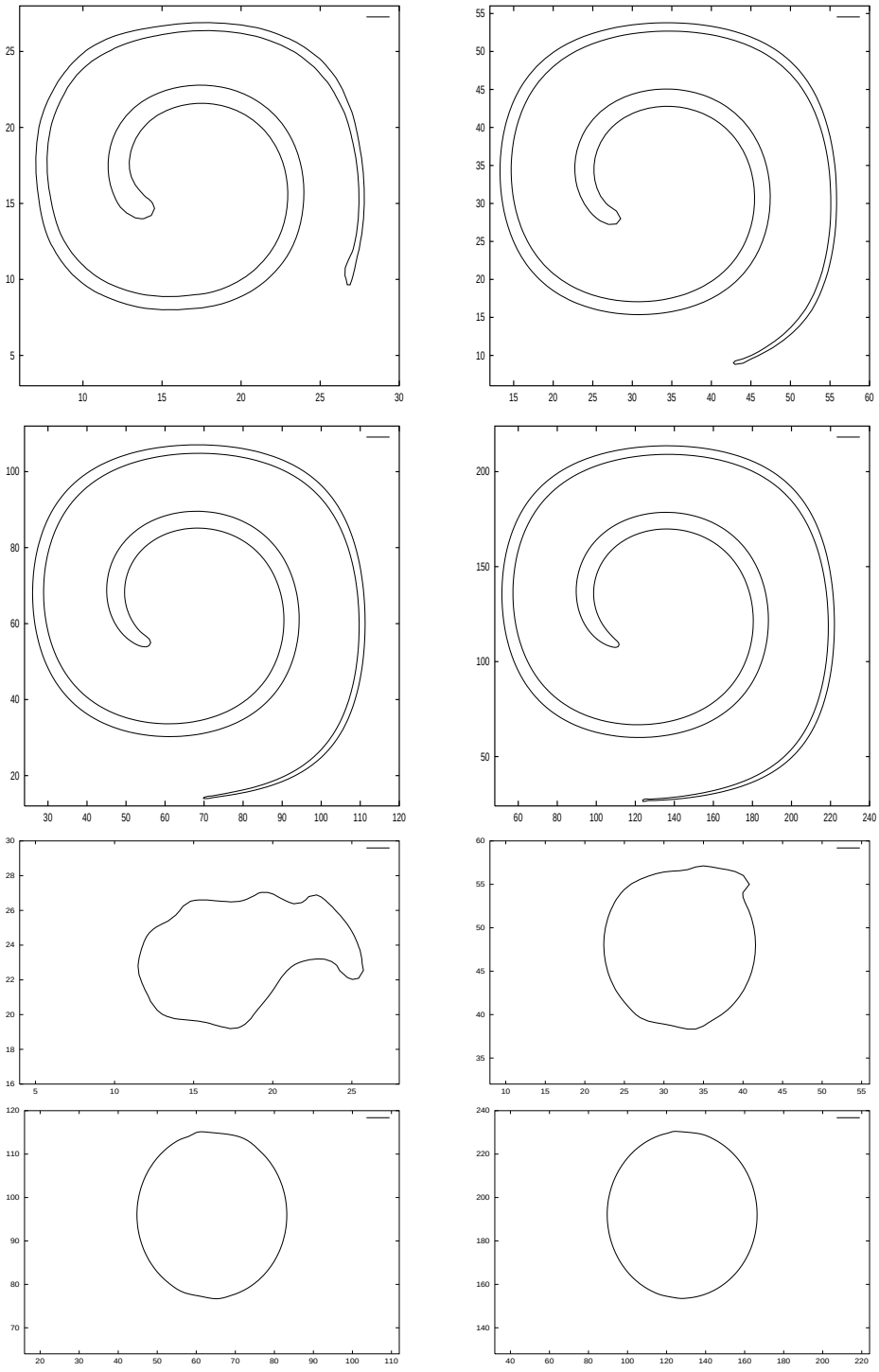

Fig. 14. The reconstructed interface at maximum deformation at $t = 4.0$ and back to the initial position at $t = 8.0$ for the single vortex field test with $T = 8$ for the following meshes with 32^2, 64^2, 128^2 and 256^2 cells (from left to right and from top to bottom, respectively).

Table 4
Mass (E_m) and geometric (E_g) errors for the single vortex test with $T = 8$, at different resolutions and *CFL* numbers

n_x	*CFL*	E_m	E_g	E_g (KR)	E_g (HF)
32	1.0	9.42e − 3	2.53e − 2	4.78e − 2	3.72e − 2
	0.1	1.65e − 3	3.45e − 2		
64	1.0	1.94e − 3	2.78e − 3	6.96e − 3	6.79e − 3
	0.1	3.14e − 4	3.75e − 3		
128	1.0	2.53e − 4	4.78e − 4	1.44e − 3	1.18e − 3
	0.1	6.60e − 5	3.21e − 4		
256	1.0	5.95e − 5	1.16e − 4		
	0.1	1.58e − 5	1.01e − 4		

The data in the fifth and sixth column are taken from [24] and [10], respectively.

algorithm simplifies considerably the structure of the interface that develops the two bulges. At higher resolution, the interface tail and head are better resolved and further stretched by the vortical flow. More importantly, the filament does not break, while in VOF/PLIC simulations the interface breaks up in high curvature regions (see for example [10,24]), which are resolved inadequately by just a segment in a cell when the radius of curvature is comparable with the grid spacing. Moreover, we notice that the method performs quite well when the fluid shape is stretched and more intersection and conservation markers are added. On the contrary, on the way back to the initial position, and in particular at low resolution, the interface line is locally compressed and the method has to deal with a great number of points in the same cell and our simple rules of markers reduction leave a short tail behind the head of the interface. On a 128^2 mesh the tail has completely disappeared. We are currently working on the markers reduction scheme to improve this feature and the results will be presented together with the three-dimensional algorithm. PLIC methods benefit a lot from the back and forth structure of the flow. The tail fragments into numerous pieces as the interface is stretched, but these coalesce on the way back. In Table 4 we compare again our results with those presented in [10,24]. As for the $T = 2$ single vortex test, the mass conservation error is constantly decreasing with the *CFL* number. The geometric error shows now a similar behavior only at high resolution. The reason is that at low resolution, when $n_x = 32, 64$, most of the error comes from the cells near the tail where the thickness is very small. These cells are crossed by the interface in several points and our simple markers reduction algorithm does not always produce a good result. Nevertheless, our hybrid scheme compares favorably with PLIC methods in terms of L_1 geometric errors. In Fig. 15 we show the evolution of the fluid shape on a 64^2 mesh with $T = 12$ till it spirals about six times around the center. We can see that the interface is still very well resolved and even if the filament is quite thin the method tracks the interface efficiently. PLIC methods cannot cope with such a filament and the interface is again broken in several pieces with an artificial topology change. We cannot compare directly our results with those obtained in [6] with a hybrid particle level set method, because of the different geometric error metric in (14) and in [6, Eq. (14)] . However, a qualitative inspection of the results obtained in the single vortex and deformation field tests shows that they compare well with the particle level set solutions of [6].

4.3. Deformation field test

An even more radical test is the deformation field where we place an initial circular shape in a vortical flow with 4×4 or 8×8 vortices. The periodic velocity field is given by the stream function [6,24,31]

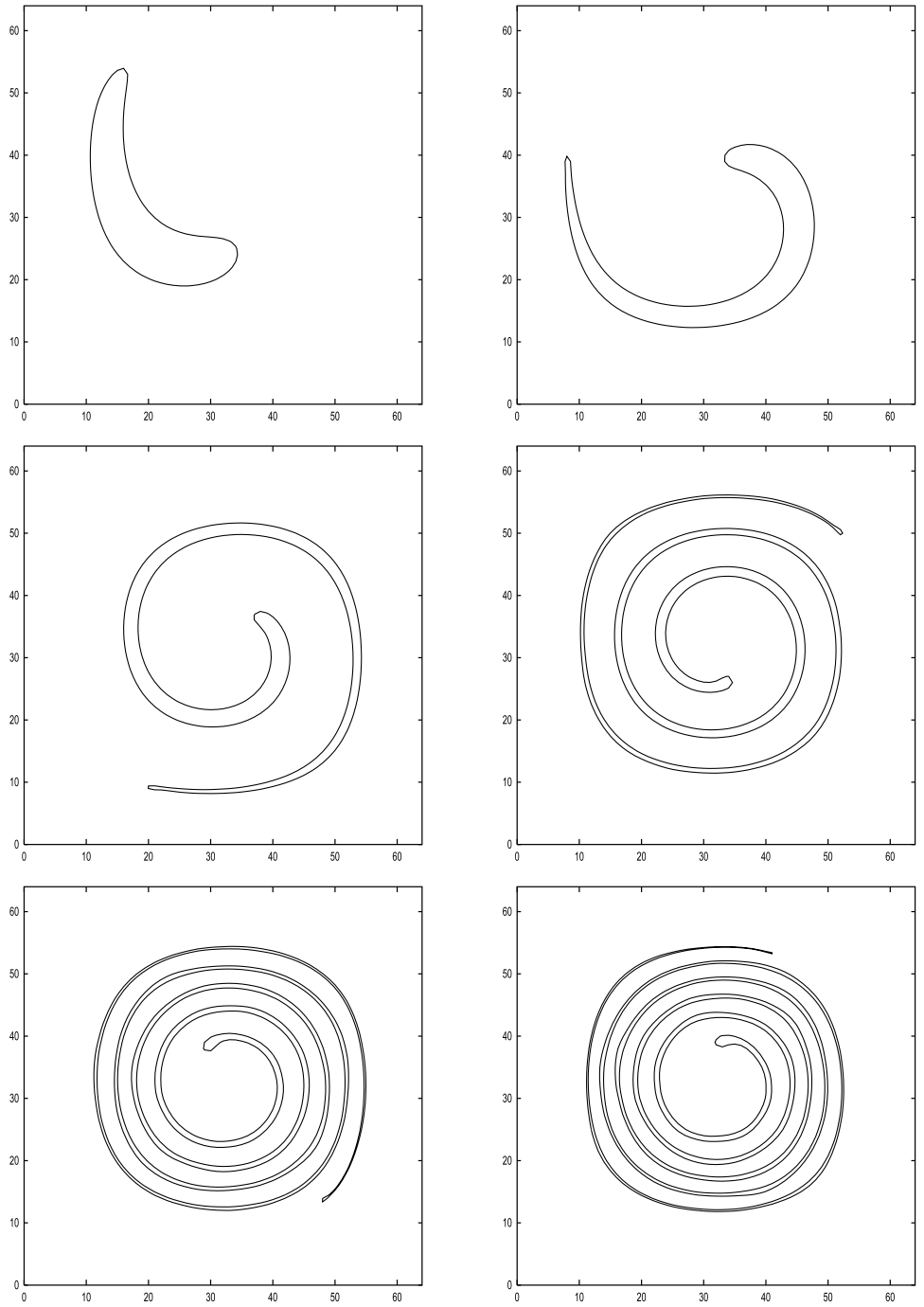

Fig. 15. The reconstructed interface at $t = 0.5$, $t = 1.0$, $t = 2.0$, $t = 4.0$, $t = 8.0$ and $t = 12.0$ for the single vortex field test with $T = 12$, on a 64^2 mesh.

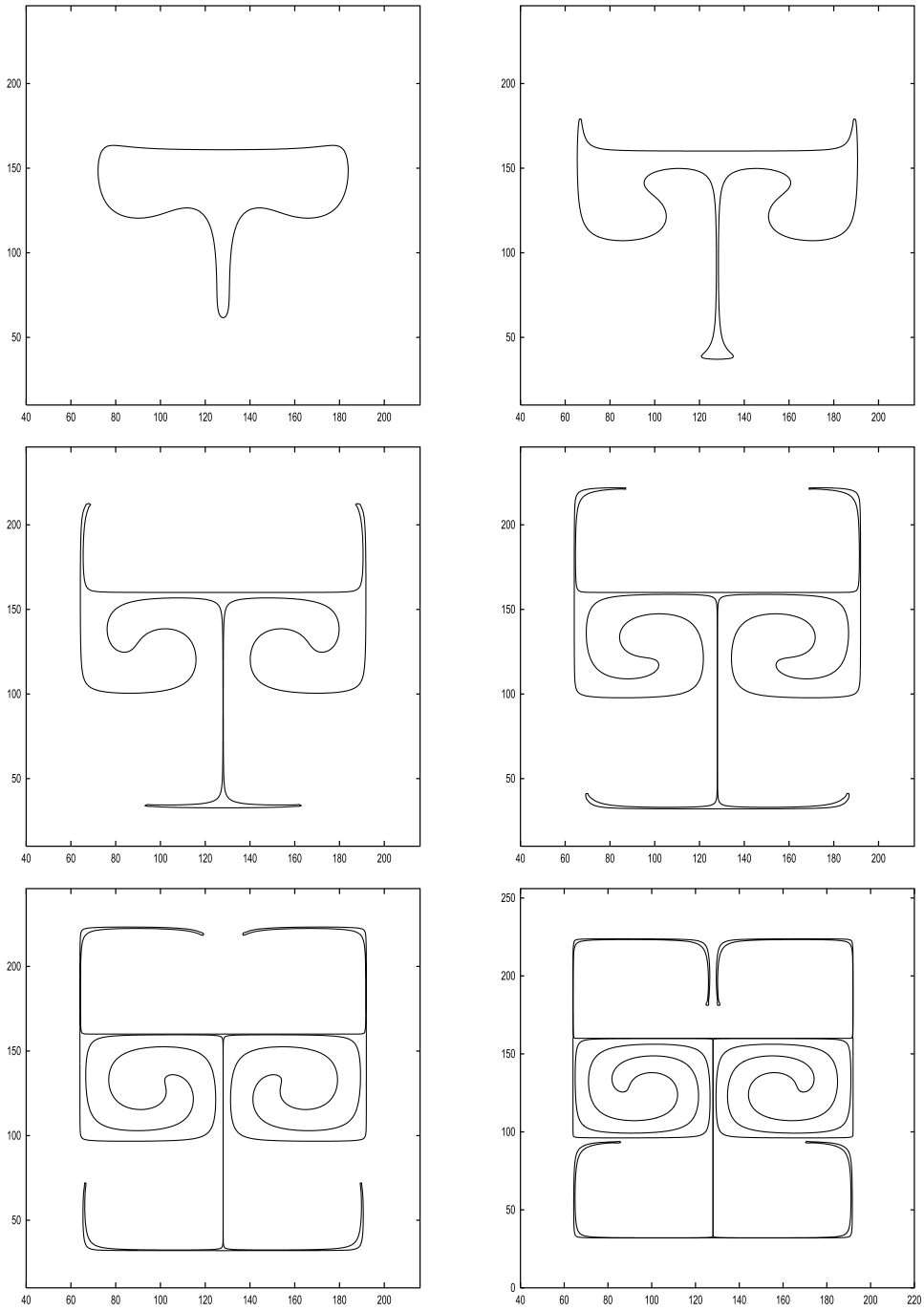

Fig. 16. The reconstructed interface at $t = 0.35$, $t = 0.7$, $t = 1.0$, $t = 1.35$, $t = 1.7$ and $t = 2.0$ for the deformation field test with 4×4 vortices, on a 256^2 mesh.

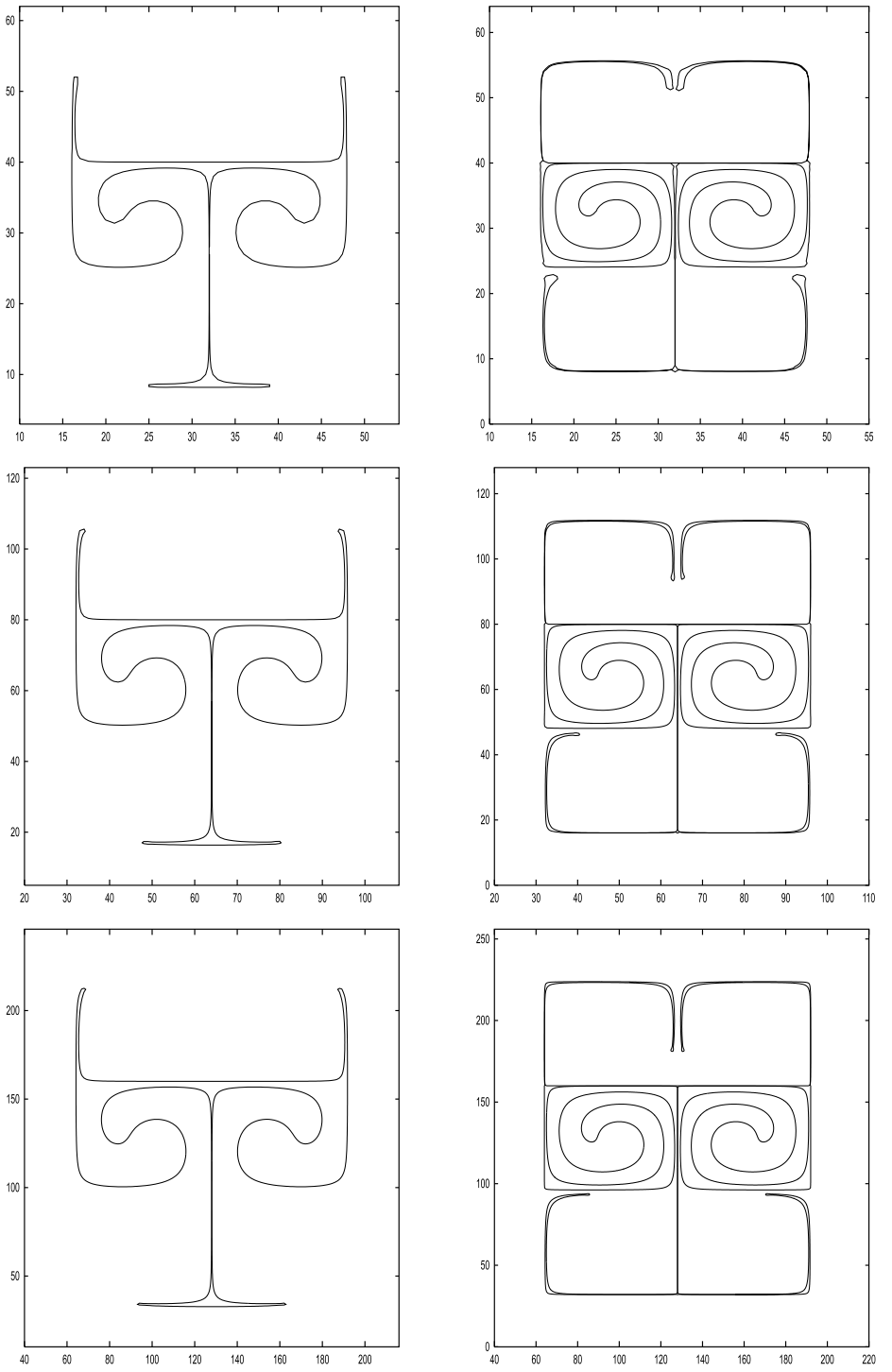

Fig. 17. The reconstructed interface at $t = 1.0$ and $t = 2.0$ for the deformation field test with 4×4 vortices, on several meshes with 64^2 cells (top), 128^2 cells (middle) and 256^2 cells (bottom), respectively.

Table 5
Mass (E_m) error for the deformation field test with 4×4 and 8×8 vortices at time $t = 2.0$

n_x	Vortices	*CFL*	E_m ($t = 2$)
64	4×4	0.1	1.03e − 4
128	4×4	0.1	5.45e − 4
256	4×4	0.1	2.23e − 5
256	8×8	0.1	3.47e − 5

$$\psi = \frac{1}{n\pi} \sin(n\pi(x + 0.5)) \cos(n\pi(y + 0.5)), \tag{17}$$

where n is an integer number that fixes the number of vortices, with $n = 4$ we have the 4×4 vortex deformation field and for $n = 8$ we have 8×8 vortices. The velocity field is normalized in such a way that its value corresponds to the local *CFL* number. The circle is now placed in the center of the domain, in the middle of several vortices, and it is progressively entrained by the vortices undergoing strong deformation with the formation of several filamentary structures, and by the time $t = 2$ the interior spiral has completed a full rotation. In Fig. 16 we show the evolution of the interface at different times up to $t = 2$ on a 256^2 mesh, while in Fig. 17 the interface is shown at times $t = 1$ and $t = 2$, but for three different meshes with 64^2, 128^2 and 256^2 cells, respectively. For short times, $t \leqslant 2$, the solution can be accurately computed and the filaments are reproduced with good resolution as shown in the previous two figures. For longer times, $t > 2$, the filaments break and the correct solution cannot be tracked any longer correctly. In all simulations the maximum *CFL* number is about 0.1 and despite the severity of the deformation the reconstructed interface bears a good resemblance to the true solution, while mass is conserved rather well, as shown in Table 5. By considering the mass error results in Table 5 and the evolution of the interface as shown in Figs. 16–18, we can say that the ability of the new hybrid method to model interfaces undergoing severe stretching and conserve mass is quite good. In particular, in Fig. 17 we notice again that the bulges at the tips of the interface are better resolved and stretched along the streamlines with grid refinement. Finally, in Fig. 18 we show the tracked interface for 8×8 vortices on a 256^2 grid. In this case, the width of some filamentary regions is smaller than the grid spacing and it would be impossible to resolve them without the information provided by the markers.

5. Conclusions

A new mixed markers and volume-of-fluid (VOF) algorithm for the reconstruction and advection of interfaces in the two-dimensional space has been presented. The interface is represented by a continuous chain of segments connecting the surface markers and the spatial distribution of the reference phase is described by the volume fraction function C. The C field is updated by a geometrical unsplit advection algorithm. In each cell a polygon, containing the reference phase and delimited by a set of points containing markers and cell vertices, is advected along the streamlines. The area contribution of the deformed polygon to different computational cells is determined and then added to get the new C field. New intersection and conservation markers are placed to describe correctly the interface and to get the proper C value inside each cell. The performance of our hybrid algorithm has been compared against those of other PLIC methods for solid body translations and rotations and for non-uniform vorticity fields as well. In all examined tests, the new approach has shown a higher accuracy, in terms of both mass and geometric errors, than the alternative algorithms. The three-dimensional implementation of the algorithm, currently under development, is also of interest for physical and engineering studies.

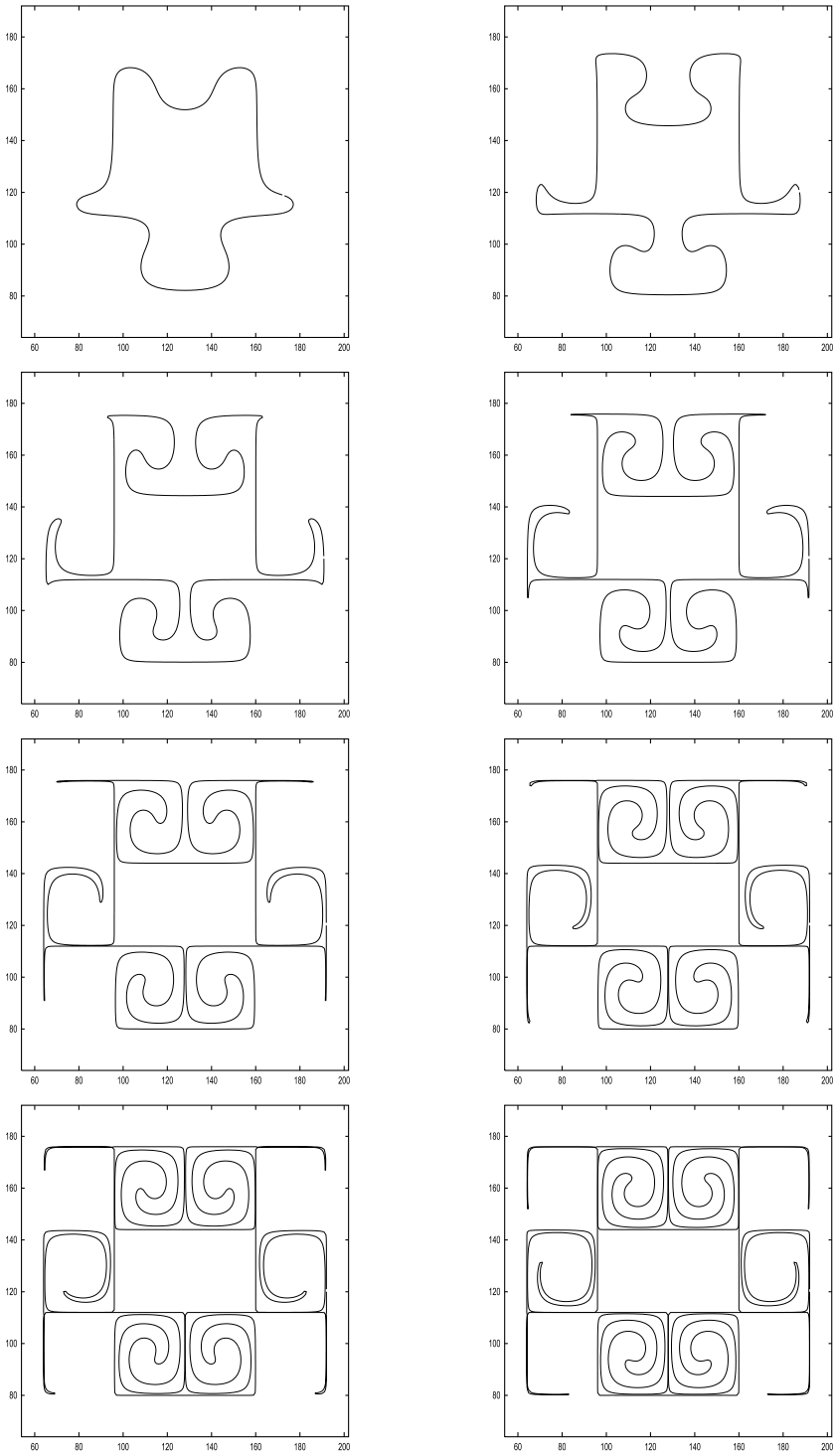

Fig. 18. The fluid body shape for $t = 0.25$, $t = 0.5$, $t = 0.75$, $t = 1$, $t = 1.25$, $t = 1.5$, $t = 1.75$ and $t = 2$ deformed by the 8×8 vortex field in a 256^2 grid.

Acknowledgements

We thank our referees for their valuable comments and helpful suggestions in completing this work.

References

[1] R. Adams, Sobolev Spaces, Academic Press, New York, 1975.
[2] A. Amsden, Numerical calculation of surface waves: a modified ZUNI code with surface particles and partial cells, Technical Report LA-5146, Los Alamos Scientific Laboratory, 1973.
[3] J.B. Bell, P. Colella, H.M. Glaz, A second-order projection method for the incompressible Navier–Stokes equations, J. Comput. Phys. 85 (1989) 257.
[4] J.U. Brackbill, D.B. Kothe, C. Zemach, A continuum method for modeling surface tension, J. Comput. Phys. 100 (1992) 335.
[5] S. Chen, D. Johnson, P. Raad, D. Fadda, The surfer marker and micro cell method, Int. J. Numer. Meth. Fluids 25 (1997) 749.
[6] D. Enright, R. Fedkiw, J. Ferziger, I. Mitchell, A hybrid particle level set method for improved interface capturing, J. Comput. Phys. 183 (2002) 83.
[7] J. Glimm, M.J. Graham, J. Grove, X.L. Li, T.M. Smith, D. Tan, F. Tangerman, Q. Zhang, Front tracking in two and three dimensions, Comput. Math. Appl. 35 (1998) 1.
[8] D. Gueyffier, A. Nadim, J. Li, R. Scardovelli, S. Zaleski, Volume of fluid interface tracking with smoothed surface stress methods for three-dimensional flows, J. Comput. Phys. 152 (1999) 423.
[9] F.H. Harlow, J.E. Welch, Numerical calculation of time-dependent viscous incompressible flow of fluid with a free surface, Phys. Fluids 8 (1965) 2182.
[10] D.J.E. Harvie, D.F. Fletcher, A new volume of fluid advection algorithm: the stream scheme, J. Comput. Phys. 162 (2000) 1.
[11] D.J.E. Harvie, D.F. Fletcher, A new volume of fluid advection algorithm: the defined donating region scheme, Int. J. Numer. Meth. Fluids 35 (2001) 151.
[12] D. Jacqmin, Calculation of two-phase Navier–Stokes flows using phase-field modeling, J. Comput. Phys. 155 (1999) 96.
[13] D. Jamet, O. Lebaigue, N. Coutris, J.M. Delhaye, The second gradient method for the direct numerical simulation of liquid–vapor flows with phase change, J. Comput. Phys. 169 (2001) 624.
[14] B. Lafaurie, C. Nardone, R. Scardovelli, S. Zaleski, G. Zanetti, Modelling merging and fragmentation in multiphase flows with SURFER, J. Comput. Phys. 113 (1994) 134.
[15] R.J. Leveque, High-resolution conservative algorithms for advection in incompressible flow, SIAM J. Numer. Anal. 33 (1996) 627.
[16] S. Osher, R. Fedkiw, Level set methods: an overview and some recent results, J. Comput. Phys. 169 (2001) 463–502.
[17] J.E. Pilliod Jr., E.G. Puckett, Second-order accurate volume-of-fluid algorithms for tracking material interfaces, Technical Report No. LBNL-40744, Lawrence Berkeley National Laboratory, 1997.
[18] S. Popinet, S. Zaleski, A front-tracking algorithm for accurate representation of surface tension, Int. J. Numer. Meth. Fluids 30 (1999) 775.
[19] W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, Numerical Recipes in C++ and C for Unix/Linux, V2.11, Cambridge University Press, Cambridge, 2000.
[20] E.G. Puckett, A.S. Almgren, J.B. Bell, D.L. Marcus, W.J. Rider, A high-order projection method for tracking fluid interfaces in variable density incompressible flow, J. Comput. Phys. 130 (1997) 269.
[21] M. Renardy, Y. Renardy, J. Li, Numerical simulation of moving contact lines using a volume-of-fluid method, J. Comput. Phys. 171 (2001) 243.
[22] W.J. Rider, D.B. Kothe, A marker particle method for interface tracking, in: H. Dwyer (Ed.), Proceedings of the Sixth International Symposium on Computational Fluid Dynamics, 1995, p. 976.
[23] W.J. Rider, D.B. Kothe, Stretching and tearing interface tracking methods, in: 12th AIAA CFD Conference, AIAA 95, 1995.
[24] W.J. Rider, D.B. Kothe, Reconstructing volume tracking, J. Comput. Phys. 141 (1998) 112.
[25] M. Rudman, Volume-tracking methods for interfacial flow calculations, Int. J. Numer. Meth. Fluids 24 (1997) 671.
[26] R. Scardovelli, S. Zaleski, Direct numerical simulation of free-surface and interfacial flow, Annu. Rev. Fluid Mech. 31 (1999) 567.
[27] R. Scardovelli, S. Zaleski, Analytical relations connecting linear interfaces and volume fractions in rectangular grids, J. Comput. Phys. 164 (2000) 228.
[28] R. Scardovelli, S. Zaleski, Interface reconstruction with least-square fit and split Eulerian–Lagrangian advection, Int. J. Numer. Meth. Fluids 41 (2003) 251.
[29] J. Sethian, Evolution, implementation and application of level set and fast marching methods for advancing fronts, J. Comput. Phys. 169 (2001) 463–502.
[30] S. Shin, D. Juric, Modeling three-dimensional multiphase flow using a level countour reconstruction method for front tracking without connectivity, J. Comput. Phys. 180 (2002) 427–470.

[31] P.K. Smolarkiewicz, The multi-dimensional Crowley advection scheme, Mon. Weather Rev. 110 (1982) 1968.
[32] M. Sussman, E. Fatemi, An efficient, interface-preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow, SIAM J. Sci. Comput. 20 (1999) 1165.
[33] M. Sussman, E. Fatemi, P. Smereka, S. Osher, An improved level set method for incompressible two-phase flows, Comput. Fluids 27 (1998) 663.
[34] M. Sussman, E. Puckett, A coupled level set and VOF method for computing 3D and axisymmetric incompressible two-phase flows, J. Comput. Phys. 162 (2000) 301.
[35] D. Torres, J.U. Brackbill, The point-set method: front tracking without connectivity, J. Comput. Phys. 165 (2000) 620.
[36] G. Tryggvason, B. Brunnen, A. Esmaeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, Y.J. Jan, A front-tracking method for the computations of multiphase flow, J. Comput. Phys. 169 (2001) 708.
[37] S.O. Unverdi, G. Tryggvason, A front-tracking method for viscous, incompressible, multi-fluid flows, J. Comput. Phys. 100 (1992) 25.
[38] M.W. Williams, D.B. Kothe, E.G. Puckett, Convergence and accuracy of kernel-basee continuum surface tension models, in: W. Shyy (Ed.), Fluid Mechanics at Interfaces, Cambridge, MA, 1999, p. 347.
[39] D.L. Youngs, Time-dependent multi-material flow with large fluid distortion, in: K.W. Morton, M.J. Baines (Eds.), Numerical Methods for Fluid Dynamics, Academic Press, New York, 1982.
[40] S. Zalesak, Fully multidimensional flux-corrected transport algorithm for fluids, J. Comput. Phys. 31 (1975) 335.