Available online at www.sciencedirect.com

SCIENCE DIRECT®

Journal of Computational Physics 197 (2004) 555–584

JOURNAL OF COMPUTATIONAL PHYSICS

www.elsevier.com/locate/jcp

A surface marker algorithm coupled to an area-preserving marker redistribution method for three-dimensional interface tracking

Eugenio Aulisa, Sandro Manservisi [1], Ruben Scardovelli *

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

Received 21 May 2003; received in revised form 10 December 2003; accepted 11 December 2003
Available online 23 January 2004

Abstract

In this work we present a new interface tracking algorithm based on surface markers to reconstruct and advect interfaces in three-dimensional space. The algorithm is coupled to a marker redistribution method, which is area-preserving along the interface. The interface is described by a set of closed lines which define a coarse Lagrangian quadrangular mesh in the fixed computational grid. Fixed markers are located where the lines cross and are maintained during the whole simulation. At each time step the interface is first reconstructed and then all markers are advected by following streamlines. In the reconstruction step, new markers are determined by computing the intersections of the interface lines with the grid cell faces and by adding area conservation markers between fixed and grid intersection markers. Intersection and conservation markers defined at the previous time step are discarded. The method maintains a smooth geometrical description of the interface for both two-dimensional and three-dimensional tests with accurate volume conservation even in very challenging situations where the fluid bodies progressively deform and stretch developing localized regions with very high curvature and thin fluid filaments. The method compares favorably with front capturing methods, such as volume-of-fluid and level set, and other hybrid techniques, such as the particle level set method.

PACS: 65C20; 68U20

Keywords: Multiphase flows; Interface reconstruction and advection; Front tracking and front capturing; Hybrid methods

* Corresponding author. Tel.: +39-51-644-1720; fax: +39-51-644-1747.
E-mail address: raus@mail.ing.unibo.it (R. Scardovelli).

[1] Present address: Department of Mathematics and Statistics, Texas Tech, Lubbock, TX, USA.

0021-9991/$ - see front matter © 2004 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2003.12.009

1. Introduction

The numerical simulation of multiphase and free-surface flows is a vast topic with a variety of applications including basic fluid mechanics, engineering and environment. Several numerical methods have been devised and used to model complex 2D and 3D flows exhibiting topology changes. Popular Eulerian front capturing schemes, where the interface is defined in some way on a fixed grid, include volume tracking or volume-of-fluid (VOF), level set and phase field methods. Some recent reviews of the capabilities and limits of these methods to model flows with large, not uniform vorticity together with an extensive bibliography can be found in [12,13,17,22,23,25]. In particular, volume tracking is a geometric method based on the volume fraction function C. Mixed cells, which are crossed by the interface, have a value of C between zero and one, the other cells have a value of C either one, in the reference phase, or zero. The *color function* C can be derived from the phase indicator function χ, equal to one in the reference phase and zero in the other phase. χ is discontinuous across the interface and the discrete value of C is obtained by integrating χ over the grid cell volume. The most recent version of volume tracking is the piecewise-linear interface calculation (PLIC) where the interface is approximated by a linear function in each mixed cell [19,22,24]. The interface is in general not continuous across neighboring cells. In the level set approach, a smooth function φ is defined to represent the interface as the set of points where $\varphi(\vec{x}, t) = 0$. The level set function will then be positive on one side of the interface, for example inside the reference phase, negative on the other one. All these methods are simple to implement since they are based on a single scalar function defined on the computational domain and do not require book-keeping and dynamical resizing of data structures containing information on the evolving interface geometry as in Lagrangian front-tracking methods. They have been used by many authors to simulate multiphase immiscible incompressible flows [8,14,19,29,30] and boiling flows as well [28,35].

To assess the capability of interface tracking methods to follow accurately the evolution of surfaces undergoing strong deformation several two-dimensional test problems have been proposed in [21,22]. A circular fluid body is placed in incompressible flows with large vortical components and is progressively deformed and stretched developing regions with high curvature and thin fluid filaments. The flow is usually time reversed to bring the fluid body back to its initial configuration. In this way it is possible to monitor the properties of the proposed method in terms of area conservation, artificial breakup and reconnection of the interface line, displacement of the fluid between the initial and final configuration, and comparison with an accurate high-resolution solution, when this is available.

Volume-of-fluid methods are designed to conserve mass in an incompressible flows, but the interface is poorly reconstructed with a straight line which flattens a local high curvature region and a developing filament is broken when his thickness becomes comparable with the grid spacing. As a result, filaments are artificially divided and coalesce in large blobs, appearing effectively as numerical surface tension dominated regions [11,21,22], which are dispersed along the actual interface tail. For the same tests, level set methods maintain a smooth interface, but suffer for excessive regularization in regions of high curvature and continuously lose mass [5,21].

Lagrangian techniques, either front-tracking schemes with surface markers [6,7,18,33,34] or marker particles with volume particles possessing an identity or *color* [4,10,20], by following the flow along characteristics maintain filamentary structures better than Eulerian methods and conserve adequately mass even without explicit volume conservation. On the other hand these schemes can be rather expensive, as the number of volume particles increases, or are associated to practical difficulties or complexities, such as the need to book-keep logical connections among surface elements in 3D for front-tracking methods. In this latter case, besides the addition or deletion of surface elements, as the interface is stretched or compressed by the flow, interface reconnections and breakups during topology changes complicate considerably the structure and efficiency of the algorithm. This fact has led a few authors to design new front-tracking methods without connectivity [26,32]. In particular, in [26] surface elements are physically, but not logically connected and are constructed on a level contour of a characteristic interface function, such as a Heaviside function. The reinitialization of the front based on the characteristic function is subject to the constraint that the total volume before and after reconstruction is the same and is performed by a user defined frequency. Interface merging and breakup is automatically performed during this reconstruction, thus it may somewhat depend on its frequency and as in Eulerian methods is still not related to physics, which may occur on a much smaller scale than the typical grid spacing of nowadays simulations.

More recently, a number of hybrid methods has also appeared (see for example [2,5,9,16,31]). In [31] a mixed Eulerian scheme combines the good mass conservation property of VOF methods with the accurate surface curvature representation via finite difference of the level set function, while in [5] Lagrangian disconnected marker particles are randomly positioned near the interface and are passively advected by the flow in order to rebuild the level set function in underresolved zones, such as high curvature regions and near filaments, where the level set approach regularizes excessively the interface structure and loses mass.

In this paper we improve considerably the performance in 2D of the mixed method presented in [2] and extend it to 3D. In that paper we explored the possibility to couple a Lagrangian front tracking technique with an Eulerian volume tracking method in order to describe the interface as a continuous line, in particular a series of segments connecting an ordered list of interface markers, and to better follow its motion. That technique dynamically added markers where the interface line was stretched and removed them when it was compressed. A simple marker reduction algorithm was implemented in the underresolved regions where the interface was cutting several times the same side of a computational cell. The line was overly simplified by losing all its detailed subgrid structure and by developing bulges near the tips of developing and receding filaments. In this paper as we follow the interface motion we do conserve the area at the cell level, but also locally along the interface line. As a result, the new algorithm is considerably faster and simpler, since only mixed cells are involved in the scheme, and outperforms considerably its first version. Many problems that affected the original version [2] are now solved, in particular multiple intersections of the interface with the same cell side, the clustering of marker particles within a cell and the corresponding treatment of subgrid underresolved geometry. Area conservation is now imposed locally along the interface line and not on the grid cell, therefore information on the location of new conservation and intersection markers are promptly available. This part of the algorithm can also be viewed as a method to redistribute markers uniformly along the interface in a way that the area is conserved, eventually by adding and removing them locally when this is required by the interface evolution. Filamentary structures can now be tracked efficiently, since the area-preserving algorithm conserves the area cut by the interface line and reset markers independently from the other intersection segments.

In 3D we consider a coarse Lagrangian quadrangular mesh that describes the interface. The mesh is defined as a set of closed lines and fixed markers are positioned on the grid nodes. The present formulation of the 3D algorithm is static, with no addition or deletion of lines and fixed markers and therefore no modelling of interface breakup and coalescence as well. Lines are advected separately by the flow and are reconstructed by adding grid intersection points where the line crosses the grid cell faces. Furthermore, during the advection conservation markers are added inside the cell to conserve the area along the line. In this way we move in space 1D objects, the lines, and conserve 2D quantities, the spanned areas, and end up with a very accurate description of the motion of a surface undergoing strong deformation with very good volume conservation. Fig. 1 shows the ability of the method to deal with several different objects, here a sphere, a cylinder and a cone inside a cubic box, in a deforming incompressible velocity field. For half a period the fluid bodies are stretched and develop thinner and thinner filaments which rotate around the box center without any coalescence or breakup, as implied by a divergence-free flow field. In the second part of the period the bodies are brought back to their initial position.

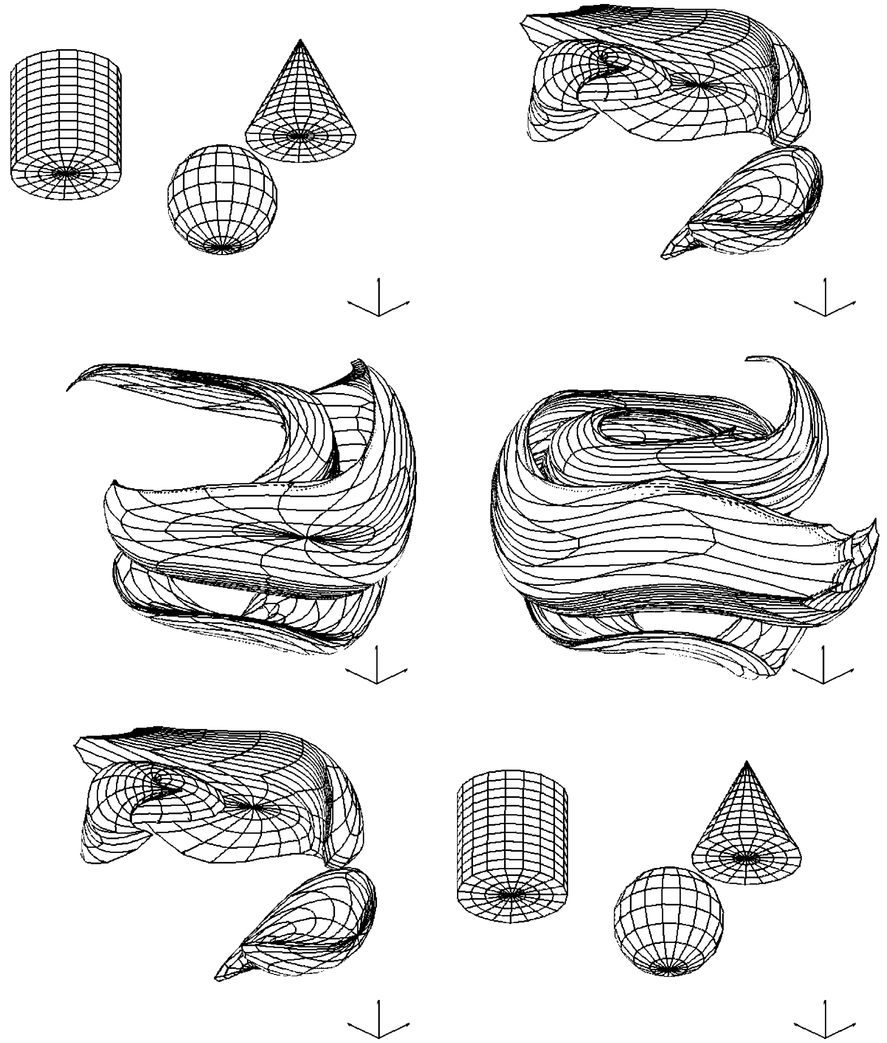

Fig. 1. Results for a sphere, a cylinder and a cone in the incompressible deformation velocity field (20) on a 32^3 grid with $T = 4$ at times $t = 0, T/10, T/5, 3T/10, T/2, T$, from left to right, top to bottom.

The remainder of this paper is organized as follows. Section 2 is devoted to the formulation of the continuous and discrete equations for interface advection. Section 3 describes the numerical algorithms for interface reconstruction and advection in 2D and 3D, while Section 4 discusses the error metrics that are used in the numerical tests. Validation tests are given in Section 5. We consider two-dimensional deforming flow fields with large vortical components (as proposed in [22]) and simple solid body translations/rotations and deforming incompressible/compressible velocity flow fields in three-dimensional geometry.

2. Continuous and discrete equations for interface advection

Let Ω be a bounded domain with the reference phase contained in the subdomain $\Omega_1 \subset \Omega \subset \Re^3$ and χ_1 the indicator function for the reference phase defined as

$$\chi_1(t,\vec{x}) = \int_{\Omega_1(t)} \delta(\vec{x}-\vec{x}')\,\mathrm{d}\vec{x}'. \tag{1}$$

The integral is over the volume $\Omega_1(t)$ bounded by the interface $S_1(\vec{x},t)$. The distribution $\delta(\vec{x}–\vec{x}')$ is a Dirac delta function that is non-zero only where $\vec{x}' = \vec{x}$, then the indicator χ_1 is one for all $\vec{x} \in \Omega_1$, zero on $\Omega - \Omega_1$ and it is discontinuous across the interface. From the phase indicator we can determine the geometric information of the interface, for example for the computation of the interface unit normal vector $\vec{n}$ we can take the gradient of (1) and transform the volume integral into an integral over $S_1(\vec{x},t)$

$$\vec{\nabla}\chi_1(\vec{x},t) = -\int_{S_1} \vec{n}(\vec{x}',t)\delta(\vec{x}-\vec{x}')\,\mathrm{d}\vec{s}', \tag{2}$$

where the normal $\vec{n}$ points into the reference phase by definition. The vector $\vec{n}$ and the curvature κ are geometric data required to compute surface tension forces and other physical interface quantities.

The fluid type does not change following the fluid paths determined by the flow field $\vec{u}(\vec{x},t)$, thus the indicator function χ_1 behaves like a passive scalar and satisfies the following advection equation

$$\frac{\mathrm{d}\chi_1}{\mathrm{d}t} \equiv \left(\frac{\partial}{\partial t} + \vec{u}\cdot\nabla\right)\chi_1 = 0. \tag{3}$$

The aim of this paper is the determination of the time evolution of the interface $S_1(\vec{x},t)$ and of the indicator function $\chi_1(\vec{x},t)$, when the velocity field $\vec{u}$ is given. From these two quantities we can evaluate the density and viscosity of the two-phase mixture, the capillary force and other singular terms defined on the interface in order to solve the single-fluid formulation of the motion equations [23]. By using the method of characteristics, the solution of (3) can be found as a function of the initial condition $\chi_{10}(\vec{x}_0,t_0)$ for all $\vec{x} \in S_1$ by solving

$$\frac{\mathrm{d}\vec{x}}{\mathrm{d}t} = \vec{u}(\vec{x},t). \tag{4}$$

The previous equation can be integrated as

$$\vec{x} = \vec{x}_0 + \int_{t_0}^{t} \vec{u}(\vec{x}(t'),t')\,\mathrm{d}t'. \tag{5}$$

If the initial position $\vec{x}_0$ of a point on the interface at time t_0 is given, then we can follow its motion by simply integrating (5). We advect all surface markers and reconstruct the fluid interface $S_1(\vec{x},t)$, then the indicator function χ_1 can be easily determined since the interface is supposed to be continuous.

Let Ω be now a bounded right hexahedron with $n_x \times n_y \times n_z$ cubic cells with grid spacing $\Delta x = \Delta y = \Delta z = h$. The reference phase is contained in the region $\Omega_{1h} \subset \Omega$ with boundary S_{1h}. The corresponding phase indicator function over the domain Ω_{1h} is denoted by χ_{1h}. 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 defined on the cell vertices or on the midpoints of the cell sides, as in a staggered MAC grid. We assume that the field $\vec{u}_h(\vec{x},t)$ can be constructed linearly from the values $\vec{u}_h$ in a finite element fashion [1], then, without restrictions, the discrete field $\vec{u}_h$ is continuous and weakly differentiable in all points.

For marker methods the appropriate equation for interface tracking is the Lagrangian formulation of (5) which we integrate to determine the evolution of the boundary $S_{1h}(\vec{x},t)$ of the subdomain Ω_{1h}. Let $\{\vec{x}_l \in S_{1h}, l = 1,2,\ldots,N_{ijk}\}$ be the marker set in the cell (i,j,k). These markers can be classified into three main categories: fixed markers N_{F}, grid intersection markers N_{I} and area conservation markers N_{C}. The fixed markers define a coarse Lagrangian surface mesh with quadrangular cells and they are gathered

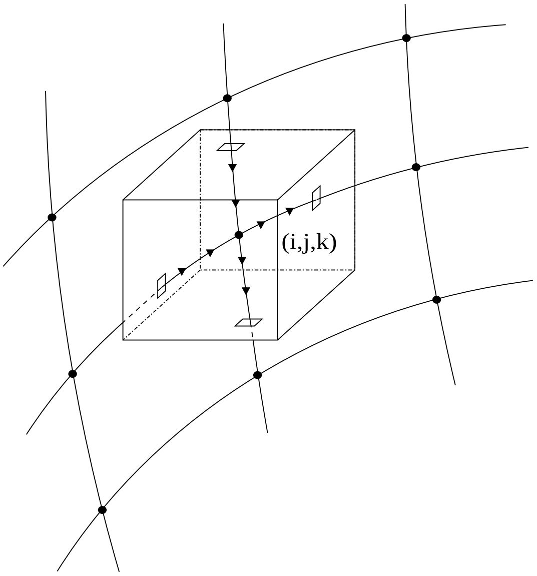

Fig. 2. The interface is represented in the cut cell (i,j,k) by fixed markers (circles), grid intersection (open squares) and area conservation (triangles) markers. The solid lines define a coarse Lagrangian quadrangular mesh on the interface.

together to define a set of closed lines. In general, a single fixed marker belongs to two different lines that cross on the marker location. Intersection markers are added at the points where the lines cross the cell faces of the computational fixed grid, while conservation markers are added inside each computational cell to conserve area and volume. In Fig. 2 we show a typical configuration around the cell (i,j,k) with a few fixed markers (circles). One of the fixed markers is inside the cell, $N_{\mathrm{F},ijk} = 1$, and the two interface lines which contain this marker cross the cell faces in four different points, where we localize the grid intersection markers (squares), $N_{\mathrm{I},ijk} = 4$. Two area conservation markers (triangles) are then added along each portion of the lines connecting the fixed marker with the intersection points, for a total of eight conservation markers, $N_{\mathrm{C},ijk} = 8$. Therefore, the total number of markers in the cell (i,j,k) is $N_{ijk} = N_{\mathrm{F},ijk} + N_{\mathrm{I},ijk} + N_{\mathrm{C},ijk} = 13$.

If $\vec{x}_l$ is one of these markers and $\vec{u}_h(\vec{x},t)$ the flow field, then we have

$$\frac{\mathrm{d}\vec{x}_l}{\mathrm{d}t} = \vec{u}_h(\vec{x}_l(t),t) \quad l = 1,2,\ldots,N_{ijk}, \tag{6}$$

for all $i = 1,\ldots,n_x$, $j = 1,\ldots,n_y$ and $k = 1,\ldots,n_z$. Given the marker position at the initial time, we can track the interface point $\vec{x}_l$ by integrating (6) 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. Numerical algorithms

In this section we describe the main features of the numerical algorithms for marker motion and interface reconstruction. We first review the two-dimensional method presented in [2] and then illustrate the changes to speed up the algorithm and improve its accuracy. Finally, we discuss the extension of the method to the three-dimensional space.

3.1. The two-dimensional method

Given the phase indicator function χ_{1h} over the discrete domain Ω_{1h} the color function $C_{ij}(t)$ is defined as

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

where A_{ij} is the area of the cell (i, j). The function $C_{ij}(t)$ represents the fraction of the cell occupied by the reference phase, assuming a value equal to one at all cells located in the interior of Ω_{1h}, zero at all external cells and between zero and one at the cells cut by the interface. The interface is represented by a continuous chain of segments connecting intersection, conservation and fixed markers. Fixed markers were not defined in the first version of the method [2]. The description of the numerical algorithm is divided in two parts: interface advection and reconstruction. In the following paragraphs we briefly describe the basic ideas developed in [2] and then we illustrate the new algorithm, by pointing out the major differences between the two versions. Only the new algorithm has been extended to the three-dimensional space.

3.1.1. 2D interface reconstruction

In the description of the interface reconstruction we refer to Fig. 3. During the previous advection step the new intersection points A and B are determined and the volume fraction C is updated. The interface reconstruction after the advection step is a simple geometric procedure that places two conservation markers, points c and d of Fig. 3(a), on the segment AB, with $Ac = Bd = cd/2$. These two points are then displaced along the normal direction to their final positions C and D, respectively. The total area $S_1 + S_2$ under the polyline $ACDB$ is equal to $C_{ij}A_{ij}$, where C_{ij} is the updated value of the color function, and with the constraint $cC = dD$ the position of the two points C and D is uniquely defined and can be calculated with straightforward geometry. In principle, it is possible to introduce more than two conservation markers and to device some other displacement technique, but we have found the previous strategy simple and accurate at the same time. At the end of the reconstruction, the interface consists of a continuous chain of segments connecting a set of intersection and conservation markers ordered along the line in the counterclockwise direction with the reference phase on the left.

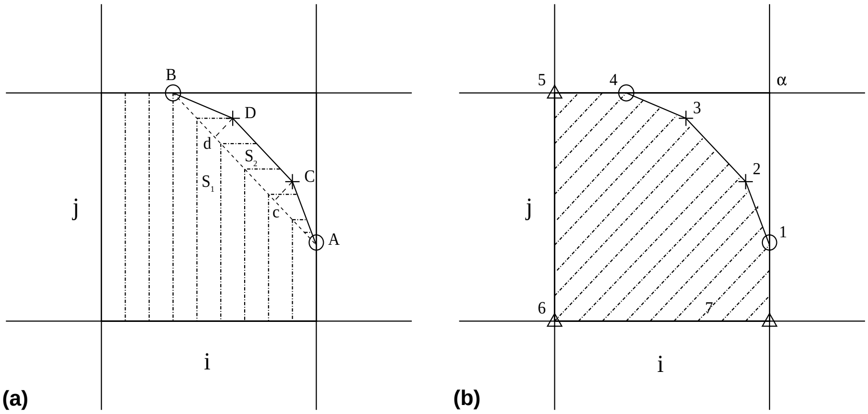

Fig. 3. Two-dimensional interface reconstruction: (a) a cut cell with two grid intersection (A,B) and two area conservation (C,D) markers; (b) the intersection points (1,4), the conservation markers (2,3) and the cell vertices (5,6,7) define a polygon containing the reference phase in its interior.

3.1.2. 2D interface advection

In the advection step we consider each cell cut by the interface and determine a polygon which contains the reference phase in its interior. The vertices of the polygon are collected in counterclockwise order. In the example of Fig. 3(b) the polygon has seven vertices: two intersection markers (circles), two conservation markers (crosses) and three cell vertices (triangles). The coordinates of the vertices are stored in normalized form. There is a local system of coordinates (ξ, η) relative to the element (i, j) and the global Cartesian system (x, y). Every point is mapped from the local system to the global system in the finite element fashion. For example the abscissa x_3 of the polygon vertex 3 inside the cell (i, j) can be determined from its local coordinates (ξ_3, η_3) in (i, j) with the standard finite element transformation

$$x_3 = x_\alpha \xi_3 \eta_3 + x_7 \xi_3 (1 - \eta_3) + x_5 (1 - \xi_3) \eta_3 + x_6 (1 - \xi_3)(1 - \eta_3), \tag{8}$$

where the x_i are the abscissae of the four vertices $\alpha, 5, 6, 7$ of the cell (i, j). To simplify the algorithm we advect along the streamlines only the four vertices of the cell to the new points $\alpha', 5', 6', 7'$ as shown in Fig. 4, with the following fourth-order Runge–Kutta scheme

$$\vec{x} = \vec{x}_0 + \left(\frac{\vec{k}_1}{6} + \frac{\vec{k}_2}{3} + \frac{\vec{k}_3}{3} + \frac{\vec{k}_4}{6} \right), \tag{9}$$

where $\vec{k}_1 = \Delta t \vec{u}(\vec{x}_0, t)$, $\vec{k}_2 = \Delta t \vec{u}(\vec{x}_0 + \vec{k}_1/2, t)$, $\vec{k}_3 = \Delta t \vec{u}(\vec{x}_0 + \vec{k}_2/2, t)$ and $\vec{k}_4 = \Delta t \vec{u}(\vec{x}_0 + \vec{k}_3, t)$. We consider a continuous spatial representation of the velocity field, but the time level is kept fixed. If the velocity field is stored for two consecutive time levels, then it is straightforward to extend the integration scheme in time between the two levels with a linear interpolation. In particular, the local velocity $\vec{u}_h(\vec{x}, t)$ in a point inside the cell is obtained with a bilinear interpolation of the components of the discrete velocity field $\vec{u}_h$. The new coordinates of the polygon vertices are calculated with expression (8) 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 determined. Clearly, only

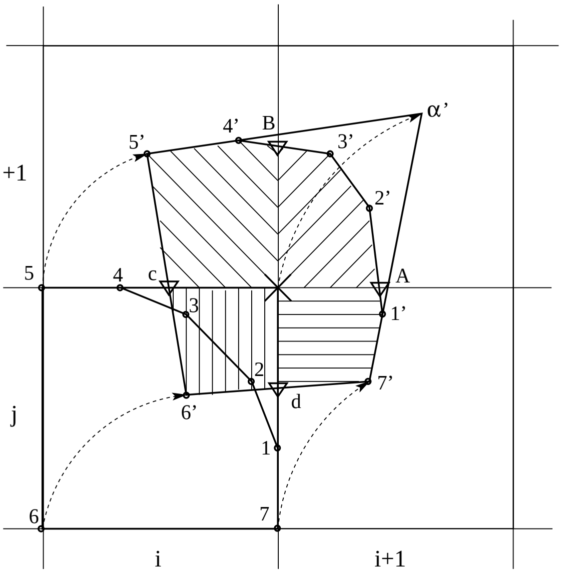

Fig. 4. Two-dimensional interface advection: the four vertices of the cell (i, j) are advected along the streamlines, the position of the points $1'$ to $7'$ and the intersections A, B, c, d with the grid lines are computed.

the two points A, B are new intersection markers. The reference phase originally inside the donor cell (i, j) after advection can be displaced over several neighboring acceptor cells. The contribution to each acceptor cell of Fig. 4 is one of the four regions with a different hatching and its area can be easily calculated, since the coordinates of the vertices of each polygon are known [2]. This procedure is repeated for all full and cut cells to get the updated color function C and all new intersection markers. Conservation markers are finally added to complete the interface reconstruction.

3.1.3. 2D area-preserving marker redistribution algorithm

In this paragraph we discuss the most important differences between the old algorithm presented in [2] and its new formulation, while the numerical comparison of the performance of the two algorithms is presented in Section 5. Many problems that were present in the previous version, such as multiple intersections over one cell side, the clustering of markers within a cell and the corresponding treatment of subgrid underresolved geometry, are now solved satisfactorily. In the old version of the algorithm, when multiple intersections with a cell side were present, we removed some of them, thus simplifying considerably the interface structure, since the constraint of area conservation in the cell was not sufficient to accurately position the line. In the new algorithm the area conservation constraint is applied locally along the interface line and therefore the location of the new conservation and intersection markers can be immediately calculated. Filamentary structures and even point crossings can be tracked efficiently, since the area-preserving algorithm conserves the area cut by the interface line and set the appropriate markers independently from the other intersection segments. To illustrate the redistribution algorithm, we consider Fig. 5 where the reference phase in the cell $(i + 1, j + 1)$ after advection is contained inside the polygon with vertices $A, 2', 3', B, \alpha$. The reconstruction step can be geometrically visualized by subdividing this polygon in $AB\alpha$ and $A2'3'B$. The first polygon remains fixed, as the area S_1 in Fig. 3(a), while in the second region we first discard points $2', 3'$ and then add points C, D, with the constraint that the area $ACDB$ is equal to $A2'3'B$. This simple consideration has several noticeable implications. We now conserve mass locally along the interface and therefore also in the cell as a whole, as a result we add and remove points on the line only where this is needed, without reducing the number of intersection markers in the case of multiple

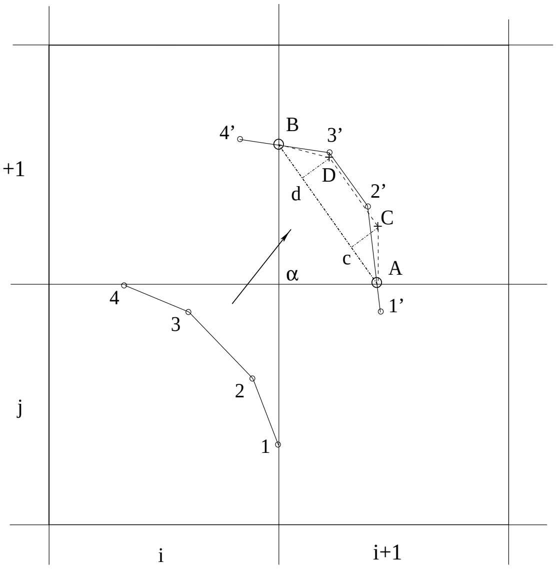

Fig. 5. Interface reconstruction algorithm: the two inner points $2'$ and $3'$ are substituted by the two conservation markers C and D.

intersections with the same cell side. The method is also much faster: we need to advect only the cut cells and not the full ones, moreover we calculate the area of polygons along the interface and not in the whole region occupied by the reference phase, as shown in Fig. 4. The new procedure is depicted in Fig. 6, where the number of markers inside the two cells $(i+1, j)$ and $(i, j+1)$ will be reduced in the interface reconstruction step.

Notice in Fig. 5 that inside each cut cell the markers are rather evenly spaced, since $Bd = Ac = 0.5cd$ by construction. The length of the interface arc in any cell can be of the order of the grid spacing h, but also much smaller, as in the cell $(i+1, j-1)$ of Fig. 6 where the interface cuts the cell boundary very close to one of its vertices. In this way, at any given simulation time we can have a not uniform distribution of markers along the interface. However, if we average in time over a characteristic cell crossing time Δt_{cr}, then the distribution becomes much more homogeneous. Furthermore the total number of markers is dynamically adjusted: markers are added as the interface line is stretched and more cells are intersected, on the contrary markers are removed when the line is locally compressed and fewer cells are crossed.

The volume fraction field C can now be calculated only after the interface has been reconstructed, by collecting in counterclockwise order all intersection and conservation markers as well as the cell corners located inside the reference phase domain Ω_{1h}. Fixed markers are not strictly necessary in 2D. However, if make use of them and if a fixed marker is located inside a cell between two intersection markers, we simply reconstruct the interface independently on each of the two segments connecting the fixed marker with the intersection points. Therefore in the cells containing a fixed marker, the number of inner points doubles, but for well-resolved interfaces we do not observe any significant change in the results. On the other hand, fixed markers are particularly helpful when the interface has corners, as discussed in Section 5.

3.2. The three-dimensional method

A 3D-surface mesh is usually described by using structured or unstructured, triangular or quadrangular meshes. We describe interfaces in 3D by a set of lines. On these lines, the fixed markers correspond to the nodes of the mesh. If the interface is covered by a triangular unstructured mesh, then it can be described by

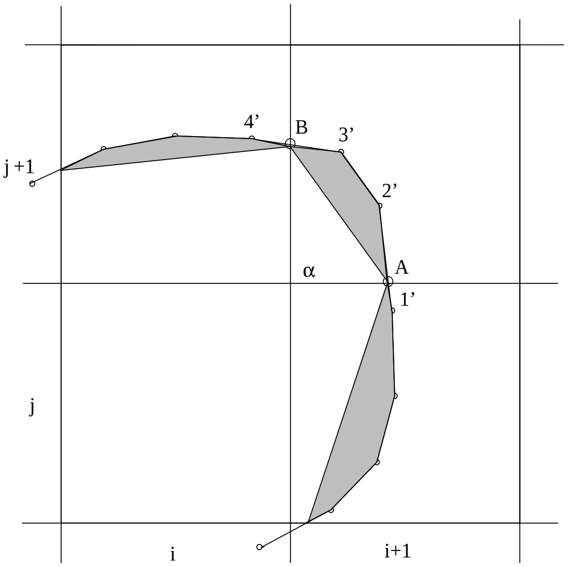

Fig. 6. Local area conservation algorithm: only the gray areas are conserved when the interface is reconstructed.

a line, or a set of lines, following the perimeter of each triangle and then moving to the neighboring element, till the whole triangulation is covered. With this strategy, a single marker may enter several times in the line. An efficient description of the interface can be obtained by minimizing the line entries of a single marker. The number of lines and the chosen topology can play a positive role in the execution time of the algorithm. However, the computational time required for the interface reconstruction and advection is usually negligible when compared to the time necessary to integrate the Navier–Stokes equations. If the interface is covered by a structured mesh, a regular triangular or quadrangular mesh can be arranged. A structured rectangular mesh can be easily generated by a set of intersecting lines. In our tests, we consider at the initial time simple solid geometric figures, such as spheres, cylinders and cones, whose surface can be easily represented by a regular mesh, and take the distance between two consecutive nodes comparable to the fixed grid spacing h. In Fig. 7 we show a possible discretization of a cylinder. The two lines a and b correspond to the cylinder top and bottom circles. When this grid is mapped onto the cylinder, the top and bottom lines degenerate into a point, the center of the two circles, however these coincident points maintain their individuality, since they belong to different lines. The initial surface grid consists of an ordered list of lines and each line of an ordered list of points. The interface advection in the given flow is reduced to the separate motion of these lines and points in the three-dimensional space. The present formulation of the algorithm is static, with no addition or removal of lines and fixed markers during the simulation. The motion of these lines describes the interface evolution and the Lagrangian surface mesh keeps its structure unchanged in time. The number of fixed markers is not sufficient either to describe correctly a fluid body undergoing strong deformation or to conserve accurately volume. Therefore intersection markers on the cell faces and conservation markers inside the cell are placed between fixed points in order to track adequately the fluid body shape and its volume. The 3D algorithm is again divided in two steps: interface reconstruction and advection.

3.2.1. 3D interface reconstruction

The Lagrangian mesh that defines the fluid body interface consists of a set of lines which are advected and reconstructed separately. Each line consists of an ordered list of points, fixed, grid intersection and area conservation markers, connected by segments. All markers are advected to their new positions which are connected consecutively by segments to design the updated interface line. The intersections between this line and the fixed grid cell boundary are then computed. The intersection points define the starting and the end points of the mesh line which lies inside a particular cell and are called grid intersection markers. The reconstruction of the interface arc inside a computational cell is then completed by deleting all non-fixed markers inside the cell and by adding new conservation markers. The procedure of removing and adding

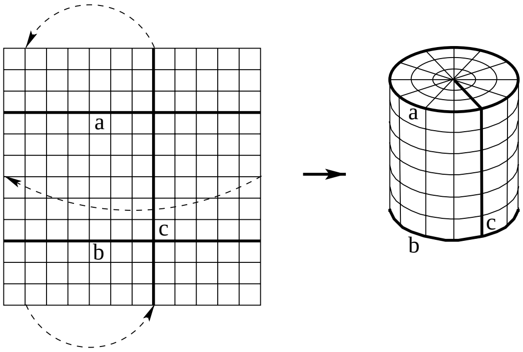

Fig. 7. Logical rectangular mesh (left) and initial surface mesh (right) of a cylindrical fluid body.

internal points is a direct extension to the three-dimensional space of the 2D technique described in the previous section and more extensively in [2]. With reference to Fig. 8, we have the two fixed points $a1, a2$, either grid intersection or fixed markers, and the three markers $b1, b2, b3$ that will be removed and substituted by the two area conservation markers $c1, c2$. The four points $a1, c1, c2, a2$ are on the same plane by construction, while the initial polyline defined by the points $a1, b1, b2, b3, a2$ does not usually lie on a plane. The final plane can be considered as an approximating osculating plane to a 3D line described by an ordered set of points connected with segments. In fact, in the limit of grid spacing $h \to 0$ the two fixed points approach each other defining the osculating plane. If the initial line arc is on a plane, its final approximation will lie on the same plane. In Fig. 8 we show how the position of the two conservation markers $c1, c2$ is computed. We consider a local coordinates system centered in $a1$ and denote with $\vec{b}$ the vector originating in $a1$ with end point b. The vector product $\overrightarrow{A1} = \overrightarrow{b1} \times \overrightarrow{b2}$ is normal to the plane defined by the three points $a1, b1, b2$ and its modulus is twice the area $A1$. Similarly, $\overrightarrow{A2} = \overrightarrow{b2} \times \overrightarrow{b3}$ and $\overrightarrow{A3} = \overrightarrow{b3} \times \overrightarrow{a2}$. The basic idea is to conserve the spanned area as we move from $\overrightarrow{b1}$ to $\overrightarrow{a2}$, through $\overrightarrow{b2}$ and $\overrightarrow{b3}$. To this aim we define the vector $\vec{A} = \overrightarrow{A1} + \overrightarrow{A2} + \overrightarrow{A3}$, which is normal to the plane containing the locally reconstructed interface and whose modulus is twice the area of the polygon with vertices $a1, c1, c2, a2$. The displacement of the two conservation markers $c1, c2$ satisfies the same constraints of the 2D problem, thus the vector $\vec{A}$ defines a unique reconstructed planar interface between the two fixed points $a1, a2$. The result of this procedure is to substitute the old intersection and conservation markers $b1, b2, b3$ with new intersection and conservation markers, $a1, a2$ and $c1, c2$ respectively. This procedure is repeated for all lines and the new interface is obtained.

The Lagrangian mesh based on the fixed markers will be locally stretched or compressed as it is advected by the flow and between two consecutive fixed markers there may be several points, either intersection or conservation markers. We use all fixed markers and a reduced number of intermediate markers to interpolate the interface away from the mesh lines. This is done only to reconstruct graphically the interface in the figures presented in the numerical tests section. Notice that once a set of interpolating functions is calculated, it is relatively easy to compute an approximate value of the volume fraction C, the unit normal $\vec{n}$ and curvature k in each cell of the fixed computational grid. To this aim we have considered two-dimensional quadratic interpolating polynomials that require eight points. In Fig. 9 we show four lines of the Lagrangian mesh that cross each other on the fixed markers (circles), several intersection and conservation markers (crosses) are also present. The set of markers for each surface element is reduced to eight points which are the basic set of points for a parabolic isoparametric finite element interpolation of the surface [1]. If the number of markers between two fixed points is odd, we take the central marker, if the number is even, we interpolate linearly between the two central points (triangles of Fig. 9). The fixed markers are mapped to

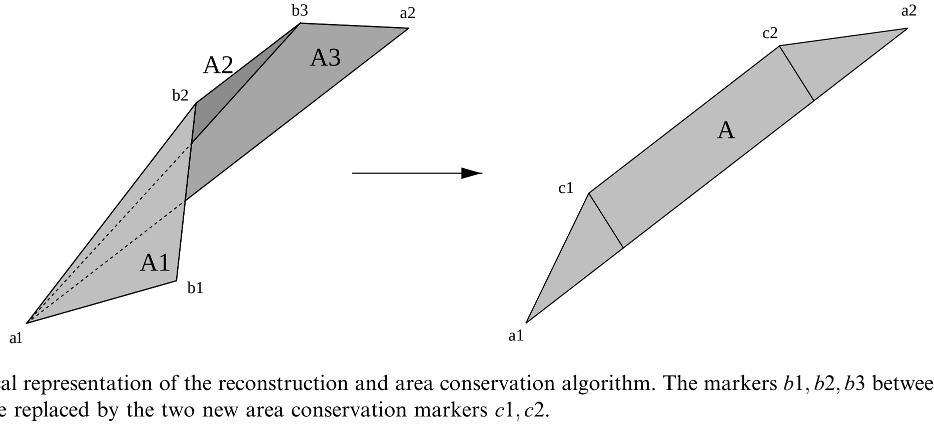

Fig. 8. Graphical representation of the reconstruction and area conservation algorithm. The markers $b1, b2, b3$ between the two fixed points $a1, a2$ are replaced by the two new area conservation markers $c1, c2$.

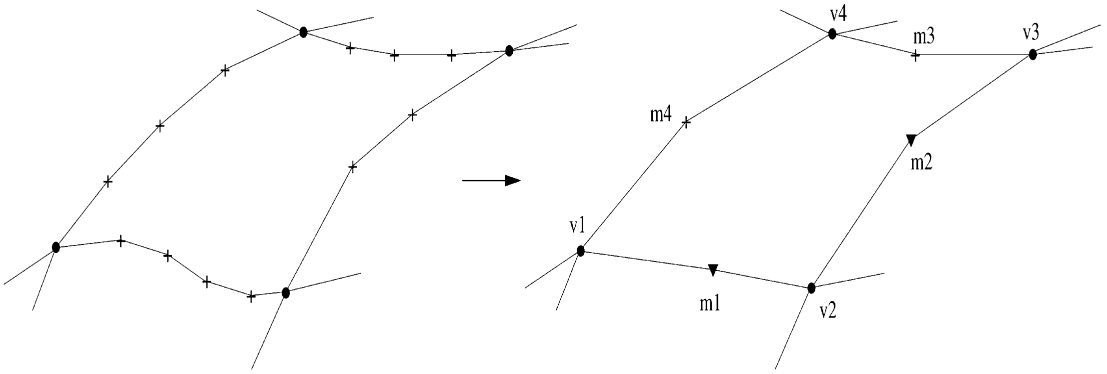

Fig. 9. The points on the boundary of a quadrangular cell of the Lagrangian surface mesh are reduced to eight for a biquadratical interpolation of the interface. If the number of points between two consecutive fixed markers is odd, the central one is considered (on the right, crosses), if it is even, the midpoint between the two central markers is taken (on the right, triangles).

the four square vertices, the other four markers to the square side midpoints. A generic point in the square has local coordinates (ξ, η), with $-1 \leqslant \xi, \eta \leqslant 1$ and its actual abscissa x is given by the expression

$$x(\xi,\eta) = \sum_{i=1}^{4} f_{vi}(\xi,\eta)x_{vi} + \sum_{i=1}^{4} f_{mi}(\xi,\eta)x_{mi}. \tag{10}$$

For the y and z coordinates the expressions are similar. The vertex basis function f_{vi} is equal to one only at the vertex vi, and the midpoint basis function f_{mi} is one at the midpoint mi. The analytical expressions of the basis functions are

$$\begin{aligned}
f_{v1} &= \frac{1}{4}(\xi^2 + \eta^2 - 1 + \xi\eta - \xi^2\eta - \xi\eta^2), & f_{m1} &= \frac{1}{2}(1-\eta)(1-\xi^2),\\
f_{v2} &= \frac{1}{4}(\xi^2 + \eta^2 - 1 - \xi\eta - \xi^2\eta + \xi\eta^2), & f_{m2} &= \frac{1}{2}(1+\xi)(1-\eta^2),\\
f_{v3} &= \frac{1}{4}(\xi^2 + \eta^2 - 1 + \xi\eta + \xi^2\eta + \xi\eta^2), & f_{m3} &= \frac{1}{2}(1+\eta)(1-\xi^2),\\
f_{v4} &= \frac{1}{4}(\xi^2 + \eta^2 - 1 - \xi\eta + \xi^2\eta - \xi\eta^2), & f_{m4} &= \frac{1}{2}(1-\xi)(1-\eta^2).
\end{aligned}$$

3.2.2. 3D interface advection

We move separately all interface lines in a Lagrangian way by advecting the markers along the streamlines, as shown in Fig. 10. We use the same Runge–Kutta scheme of the 2D case, with a continuous spatial representation of the velocity field and a fixed time level. We usually consider a staggered MAC grid with each component of the velocity field defined in the center of a different face of the cubic grid cells. The velocity field at any point inside the computational domain is obtained by a trilinear interpolation of the values of the discrete velocity field at the eight surrounding nodal points. As an example, the component u of the velocity along the x-direction is computed by

$$u(\xi,\eta,\varsigma) = \sum_{i=1}^{8} u_i f_i(\xi,\eta,\varsigma), \tag{11}$$

where u_i are the x-components of the discrete velocity field. The variables $\xi, \eta, \varsigma \in [0,1]$ are the local point coordinates with respect to the surrounding nodal points and the basis functions $f_i(\xi,\eta,\varsigma)$ are

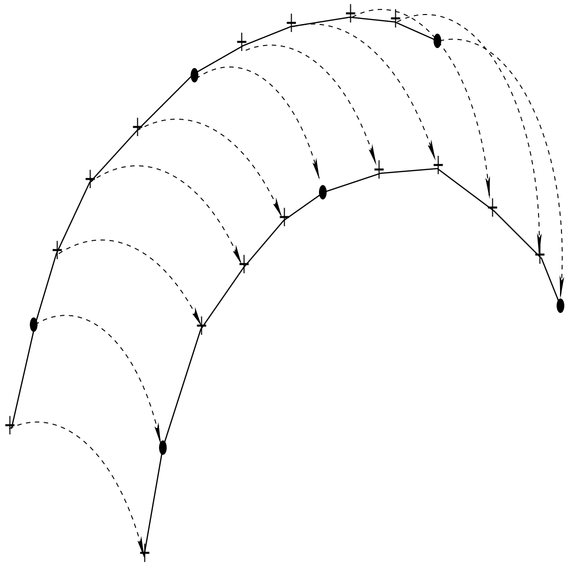

Fig. 10. All markers on an interface line are advected along the streamlines to their updated position at the next time step.

$$
\begin{aligned}
f_1 &= (1-\eta)(1-\xi)(1-\varsigma),\\
f_2 &= \eta(1-\xi)(1-\varsigma),\\
f_3 &= \eta\xi(1-\varsigma),\\
f_4 &= (1-\eta)\xi(1-\varsigma),\\
f_5 &= (1-\eta)(1-\xi)\varsigma,\\
f_6 &= \eta(1-\xi)\varsigma,\\
f_7 &= \eta\xi\varsigma,\\
f_8 &= (1-\eta)\xi\varsigma.
\end{aligned}
$$

After the line has been advected, new grid intersection and area conservation markers are computed, as described in the previous section, and the old ones are discarded.

4. Error metrics

In two-dimensional geometry we consider two L_1 error norms that are based on the color function C and are widely used. The first one is the relative area error $E_{\mathrm{a}}(t_1)$ between the total area occupied by the reference phase at the initial time t_0 and that of the deformed body at a different time t_1

$$E_{\mathrm{a}}(t_1) = \frac{|\sum_{ij} A_{ij}C_{ij}(t_0) - \sum_{ij} A_{ij}C_{ij}(t_1)|}{\sum_{ij} A_{ij}C_{ij}(t_0)}. \tag{12}$$

The latter one is the geometrical error $E_{\mathrm{g}}(t_1)$ between the position of the fluid body at the initial time t_0 and at time t_1 defined as

$$E_{\mathrm{g}}(t_1) = \sum_{ij} A_{ij}|C_{ij}(t_0) - C_{ij}(t_1)|. \tag{13}$$

Since the grid spacing h is constant, the above expressions can be simplified with the substitution $A_{ij} = h^2$. The geometric error has units of area in 2D and it is particularly useful to infer convergence rates when the fluid body deforms and returns at time t_1 to its initial position [22]. However, this definition is dependent from the fluid body area.

There are very few papers discussing quantitatively 3D reconstruction and advection algorithms, therefore we take the quality indices by extrapolating the most common two-dimensional error measures found in the literature (see for example [22]) and by taking into account directly the indicator function χ_{1h}. The relative volume error is defined by

$$E_{\mathrm{v}}(t_1) = \frac{|\int_\Omega \chi_{1h}(\vec{x}, t_0)\mathrm{d}\vec{x} - \int_\Omega \chi_{1h}(\vec{x}, t_1)\,\mathrm{d}\vec{x}|}{\int_\Omega \chi_{1h}(\vec{x}, t_0)\,\mathrm{d}\vec{x}}. \tag{14}$$

This definition is equivalent in 2D to (12) by considering expression (7) for the color function. Furthermore, we define two geometrical errors $E^i_{\mathrm{g}}(t_1)$ between the position of the fluid body at the initial time t_0 and at final time t_1

$$E^1_{\mathrm{g}}(t_1) = \int_\Omega |\chi_{1h}(\vec{x}, t_0) - \chi_{1h}(\vec{x}, t_1)|\,\mathrm{d}\vec{x} \tag{15}$$

and

$$E^2_{\mathrm{g}}(t_1) = \frac{\int_\Omega |\chi_{1h}(\vec{x}, t_0) - \chi_{1h}(\vec{x}, t_1)|\,\mathrm{d}\vec{x}}{\int_\Omega \chi_{1h}(\vec{x}, t_0)\,\mathrm{d}\vec{x}}. \tag{16}$$

The error $E^1_{\mathrm{g}}(t_1)$ is in agreement with the two-dimensional definition (13), thus it depends from the volume V_0 occupied by the fluid at the beginning of the simulation and it has units of volume. The error $E^2_{\mathrm{g}}(t_1)$ may be used in alternative to E^1_{g} in order to have a geometric error indicator that is independent from V_0. The 3D errors deserve some further discussion, in particular on how we calculate them. In the next section we consider several initial solid geometric figures, such as spheres, cylinders and cones, and approximate the actual curved surface with portions of planes, either triangles or quadrilaterals. The fixed points are located on the vertices of these geometric elements. The volume V_0 is the volume within the initial approximated surface. The fixed points do not change during the simulation, so that a final error based on χ_{1h} and the fixed points alone will not test the accuracy of our advection and reconstruction algorithm, but simply how well a fourth-order Runge–Kutta scheme can follow a particle in a continuous velocity field. We need to consider more points on the interface and, as introduced in the previous section, we may consider intermediate points between two consecutive fixed markers for a parabolic interpolation of the surface. The same set of points is involved in the calculation of both volume and geometric errors. Initially they are located exactly on the segment midpoint between two fixed markers but, as the simulation goes on, they move around because of the reconstruction and advection algorithms and at the end of the simulation they will not return to their original position. We consider all fixed markers and midpoints and calculate the volume and geometric errors as the sum of the volumes of many tetrahedra. For example, to determine the volume V_0 comprised within the interface we subdivide each surface cell in six consecutive triangles and then connect the vertices to a point inside V_0, in the case of the sphere its center, and compute the volume of the resulting tetrahedra with straightforward vector operations. In Fig. 11 we show a surface element with four fixed markers and four midpoints, including one of the six tetrahedra. The geometric error and the volume between the two interfaces at times $t = 0$ and $t = T$, when the fluid body should be back to its initial configuration, are computed in a similar fashion.

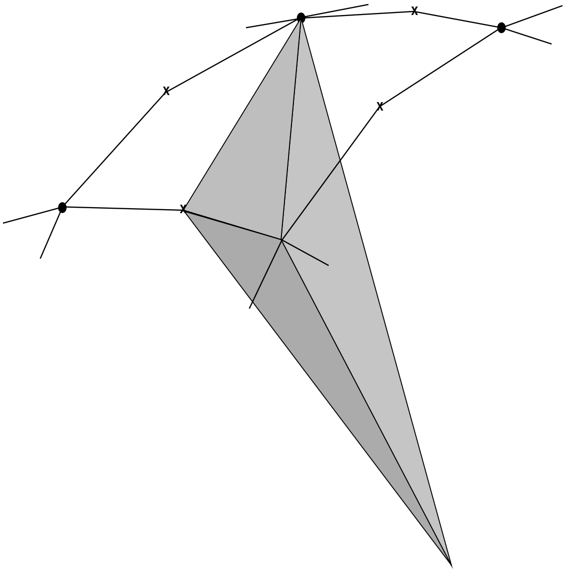

Fig. 11. To compute volume and geometric errors a surface element is defined by four fixed markers (circles) and four midpoints (crosses). The element is divided in six consecutive triangles connected to a fixed point inside the fluid body, thus defining six tetrahedra. The sum of the tetrahedra volumes of all surface elements is the fluid volume inside the interface.

5. Numerical tests

In this section we analyze the performance of our coupled marker and local area conservation algorithm over 2D and 3D tests. Translation and rotation tests in 2D have been already discussed in [2]. Mass error for such non-deforming fields is always zero. Moreover, if we consider fixed markers as well, the geometric error is much smaller, as a matter of fact it is zero if they are initially located on the grid lines. The main reason for this improvement is that fixed markers return to their initial position and the connecting segments are not deformed in these solid body motions. In 2D we discuss results about the so-called single-vortex or “vortex-in-a-box” test [3,22] and the deformation field with 16 vortices [22,27]. These velocity fields are usually modulated in time with a cosinusoidal time dependence to bring the fluid body back to its initial position [15], testing the ability of the proposed method to follow a stretching interface that develops thin filaments. A comparison between the first version of our algorithm and other tracking techniques, such as VOF, for both deforming and non-deforming velocity fields has been given in [2]. In 3D we consider the performance of our method first for uniform translations and rotations and then for a three-dimensional incompressible deformation flow field. We close the section with a short discussion about the use of this coupled technique for interface tracking in compressible kinematical velocity fields.

5.1. Two-dimensional single vortex test

The velocity field in the unit domain is defined by the stream function

$$\psi = \frac{1}{\pi}\sin^2(\pi x)\sin^2(\pi y), \tag{17}$$

with $\vec{u} = (u, v)$, $u = \partial\psi/\partial y$ and $v = -\partial\psi/\partial x$. The stream function is time reversed by multiplying it by $\cos(\pi t/T)$, where T is the period. The vortex velocity field is centered in the box with the largest velocity located half a way from the box center and the domain walls. In this velocity field we consider a circle of radius $r = 0.15$ and a square of side $l = 9/32$ both centered at $(0.5, 0.75)$. In Fig. 12 we compare the evolving interface with the method presented in [2] and the novel reconstruction method. The first one is based on the volume fraction C while the local area conservation algorithm is used in the latter approach. In particular, we show the interface at times $t = T/2$ and $t = T$, when $T = 8$ for the two grid resolutions $n = 32, 256$, while in Table 1 we compare the area and geometric errors for the two algorithms at different CFL and grid resolutions. At low resolution ($n = 32$) with the first method the head and tail of the spiral are not well resolved even during the stretching part of the motion, because of the interface local high curvature. In these two regions there are a few cells with several intersections and the marker reduction algorithm, which was used only in [2], simplifies considerably the interface structure by developing a local swelling. In the second half of the period the interface line is compressed back to its initial shape and a short tail develops that lags the real solution. We recall that the tail disappears with grid refinement. With the new reconstruction algorithm both the bulges and the tail are not present any more even at low resolution and the area and geometric errors show a remarkable improvement. In Table 1 the errors decrease both in terms of absolute values and as the CFL number tends to zero. Consistently with this picture, the area error evolution in all simulations does change with the time step and it is almost negligible around $T/2$ when the velocity field is about zero.

The coupled marker and local area conservation method is able to follow the thin filament through many rotations (more than 10 on a 32^2 without any interface breakup). We can compare our solution with the solutions given in [5]. The interface depicted in Fig. 13 rotates about four times around the center and locates itself at about the same position in Fig. 21 of [5] (actually our flow is in the counterclockwise direction) determined by a high-resolution front-tracked solution. Our method compares very well with the front-tracked result, while the hybrid particle level set solution of [5] presents a numerical surface tension dominated tail, about half a rotation in length, where the interface has broken in several pieces. By contrast, the level set solution in this test has lost a lot of mass [5], while VOF/PLIC methods, striving to conserve mass exactly, displace the fluid with less accuracy along the tail and exhibit larger and more separated blobs in underresolved regions [11,22]. Our solution does not break even on a 32×32 grid, and the head and tail of the spiral are only slightly more stretched and better resolved by increasing the grid resolution to 128 cells along each coordinate direction. We can also compare indicatively the number of markers and particles in the two methods. At the maximum deformation of Fig. 13 we have 1397 markers (this value is the sum of fixed, grid intersection and area conservation markers) at the lowest resolution and 5813 markers on a 128^2 mesh. If we mark all cells near the interface within a distance of three grid cells and consider 16 particles per marked cell [5], then we end up with 13312 and 127872 particles respectively.

We have also tested the solution sensitivity to the initial position of fixed markers when the interface line presents corners at the beginning of the simulation. In Fig. 14 on the left, at the beginning of the simulation the fixed markers are positioned on the corners and randomly one in each cut cell, on the right the fixed markers on the corners have been moved to the points where the grid lines intersect the boundary of the cells containing a corner. Even on a 32×32 grid it is rather difficult to notice any difference between the two reconstructed interfaces at the maximum stretching ($t = 4$). At time $t = T = 8$ when the fixed markers are exactly on the corners the interface matches closely the initial configuration, in the other case the corner distortion still remains localized in a couple of grid cells. This suggests that if the fixed markers can be positioned close to the singular points, for example close to droplet and bubble merging/breakup areas, the method should be able to reproduce the interface evolution very accurately. Finally, we point out that if we set initially 36 fixed markers then at $t = T/2$ there are 267 grid intersection points and 557 area conservation markers. At $t = T$ these two numbers are down to 35 and 114, respectively.

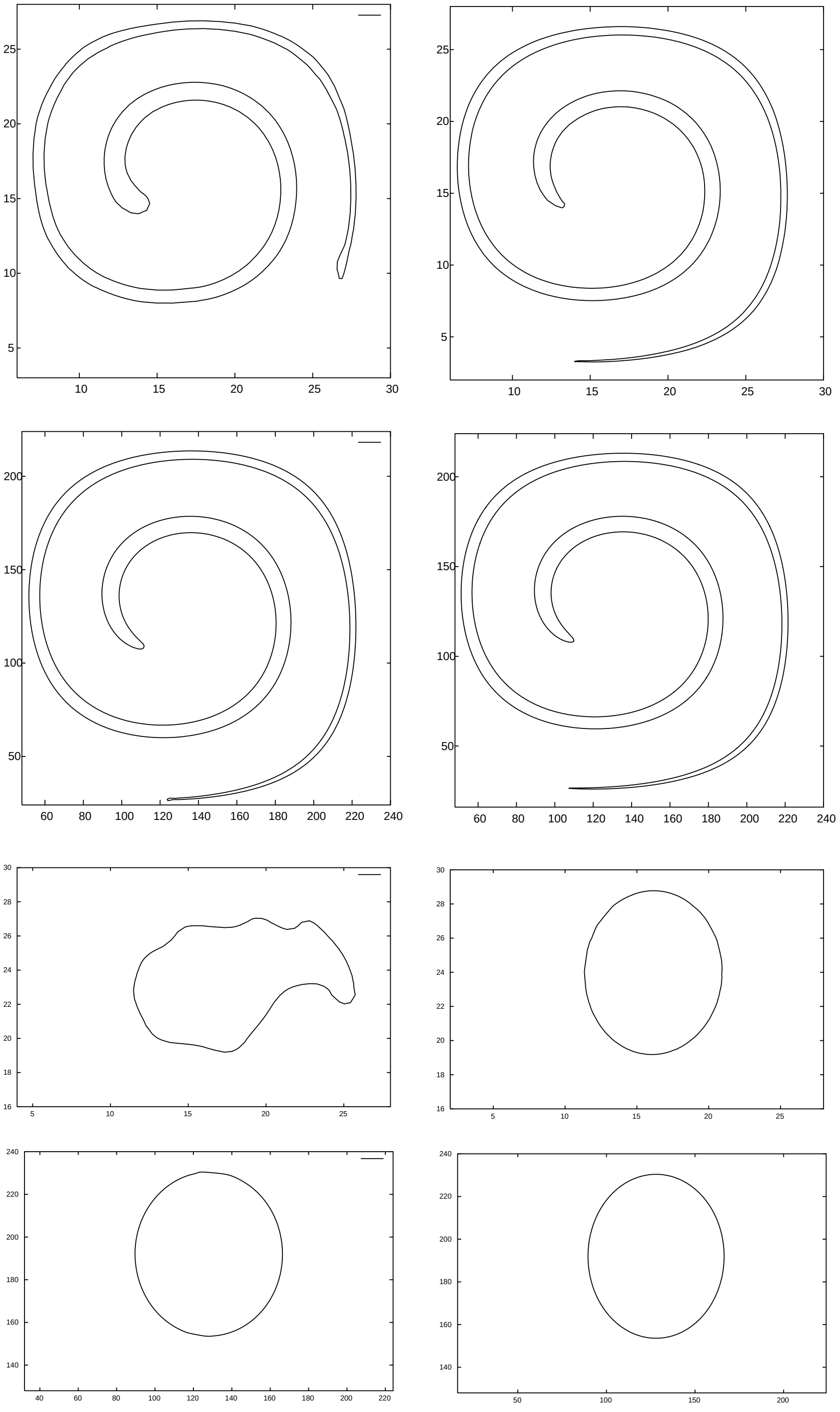

Fig. 12. The reconstructed interface at maximum deformation at $t = 4.0$ (set of top four figures) and back to the initial position at $t = 8.0$ (set of bottom four figures) for the single vortex field test with $T = 8$. For both sets, the grid has 32^2 cells on the top two pictures and 256^2 in the bottom two. On the left the results are obtained with the marker and VOF algorithm, on the right with the coupled marker and local area conservation method.

Table 1
Area (E_a) and geometric (E_g) errors for the single vortex test with $T = 8$, at different grid resolutions and CFL numbers with two different reconstruction algorithms: markers and volume fraction C (old) and markers and local area conservation (new)

n	CFL	E_a (old)	E_g (old)	E_a (new)	E_g (new)
32	1.00	9.42e−3	2.53e−2	4.85e−3	2.52e−3
	0.10	1.65e−3	3.45e−2	4.93e−4	3.09e−4
64	1.00	1.94e−3	2.78e−3	5.95e−4	3.23e−4
	0.10	3.14e−4	3.75e−3	6.23e−5	3.99e−5
128	1.00	2.53e−4	4.78e−4	7.38e−5	4.06e−5
	0.10	6.60e−5	3.21e−4	7.94e−6	5.19e−6
256	1.00	6.99e−5	1.36e−5	9.27e−6	5.11e−6
	0.10	1.22e−5	3.27e−6	1.21e−6	7.99e−7

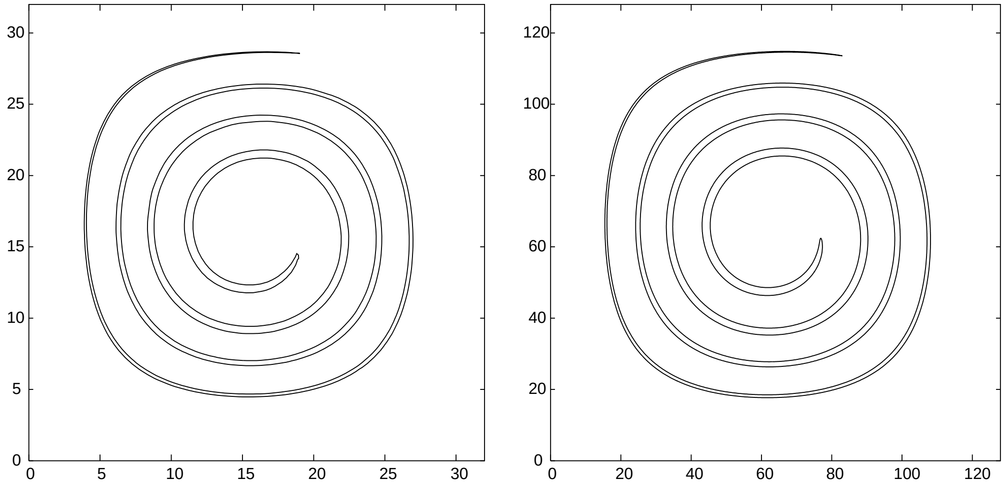

Fig. 13. A circular fluid body is placed in the single-vortex flow field and spirals about four times around the vortex center on a grid with 32^2 (left) and 256^2 (right) cells.

5.2. Two-dimensional deformation field test

This test introduced in [27] has been extensively used to study the performance of a tracking scheme when a circular fluid body is deformed by a vortical flow with many vortices. The periodic velocity field is given by the stream function

$$\psi = \frac{1}{n\pi} \sin(n\pi(x + 0.5)) \cos(n\pi(y + 0.5)), \tag{18}$$

where n is the number of vortices along any coordinate direction. In [2] a circle with radius $r = 0.15$ is centered at $(0.5, 0.5)$ and we have considered the two cases with $n = 4$, 8 and presented results only for the first half of the simulation when the interface is deformed and stretched. The results with the marker and VOF algorithm have the same features discussed in the previous section: the tips of the developing filaments develop bulges which are further stretched by the flow and better resolved by increasing the grid resolution. On the way back, when the interface line is compressed and reduced in length, the fluid

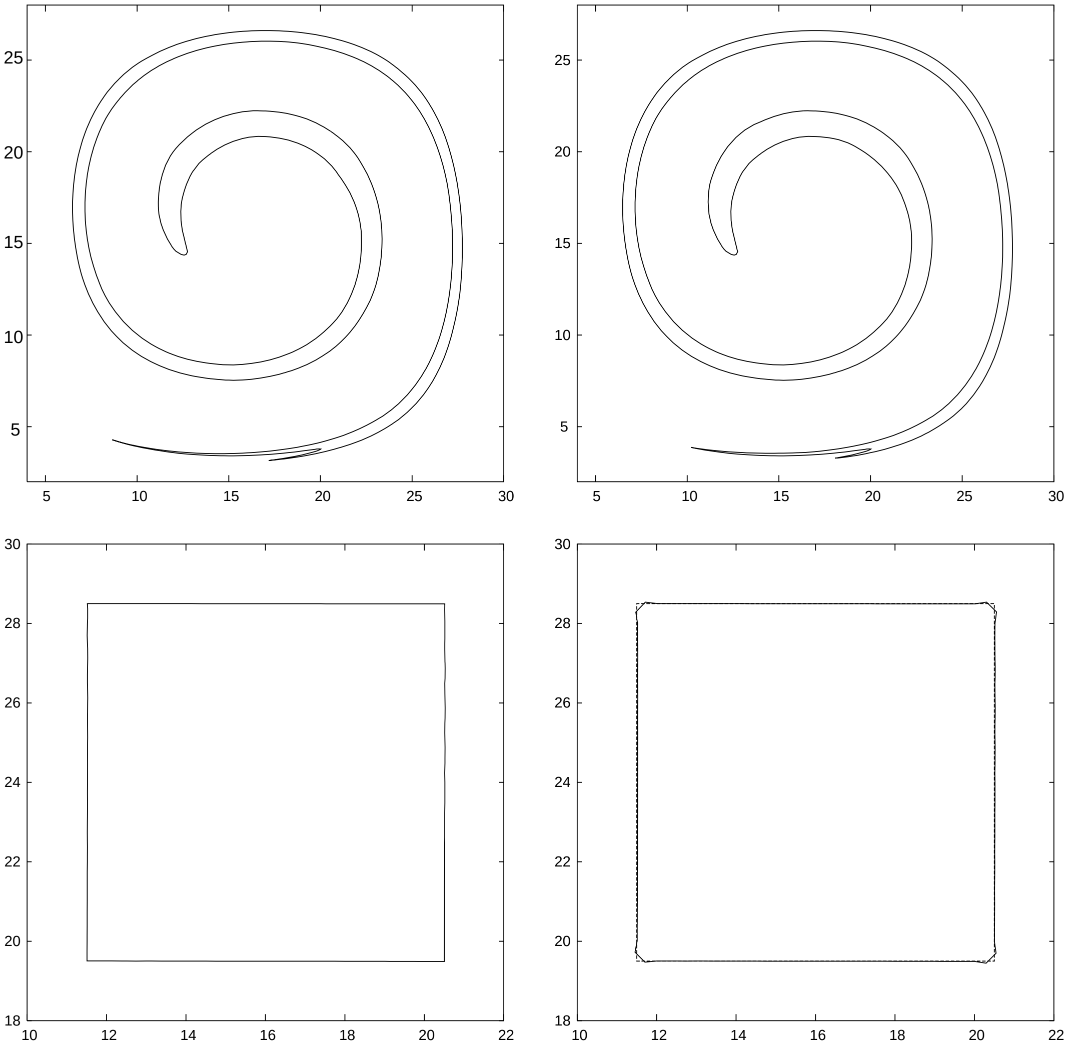

Fig. 14. A square fluid body is placed in the single-vortex flow field on a 32^2 grid. Results are shown at maximum deformation at $t = T/2$ and superimposed at times $t = 0$ (dashed line) and $t = T = 8$ (solid line). On the left the fixed markers are exactly on the corners, while on the right, in the cells containing the corners at $t = 0$, they are positioned where the square sides intercept the grid lines.

body develops a tail along each filament. Here, for comparison with other methods the stream function has again a cosinusoidal time dependence with period $T = 2$. In Fig. 15 we show the interface evolution at $t = kT/8$ $(k = 1, 2, \ldots)$ on a 64×64 grid. On this scale the two interface lines at times $t = 0, T$ cannot be distinguished and the other figures are symmetric with respect to time $t = T/2$. In Table 2 the convergence of area and geometric errors is analyzed as a function of the CFL number and grid refinement. The solution with the coupled marker and local area conservation method is very close to the high-resolution front-tracked solution of [5] and compares favorably with the results obtained by using a hybrid particle level set method and a standard level set approach in [5] and with a VOF/PLIC method in [22].

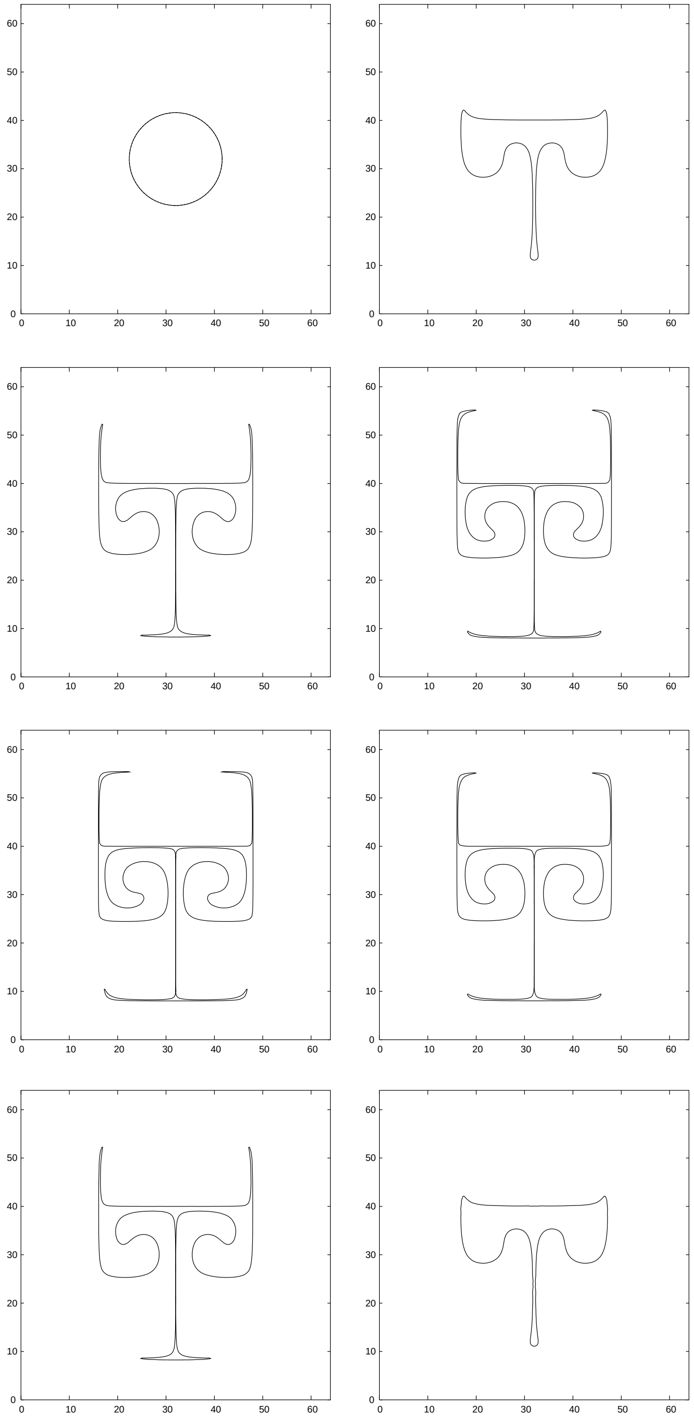

Fig. 15. A circular fluid body is placed in the unit box center in the deformation field and progressively spirals and develops long thin filaments. The results are for a 64^2 grid with a period $T = 2$. From left to right and top to bottom, the solution is shown superimposed at times $t = 0, T$ and separately at the different times $t = T/8, T/4, 3T/8, T/2, 5T/8, 3T/4, 7T/8$.

Table 2
Area (E_a) and geometric (E_g) errors for the deformation field test with $T = 2$, at different grid resolutions and CFL numbers

n	CFL	E_a	E_g
64	1.00	2.32e–4	6.82e–5
	0.10	2.73e–5	1.14e–5
128	1.00	2.26e–5	9.45e–6
	0.10	4.12e–6	1.65e–6
256	1.00	2.63e–6	1.19e–6
	0.10	4.81e–7	2.27e–7

5.3. Three-dimensional translations and solid body rotations

We first consider flows that do not induce an interface distortion and where the markers advection along the streamlines can be calculated very accurately with a fourth-order Runge–Kutta time integration scheme. The fixed markers are initially determined for all fluid bodies as in the cylinder of Fig. 7, by mapping a rectangular mesh onto a two-dimensional closed surface. They are located on the surface of the fluid body, and the curved interface is approximated by a set of connected plane quadrilaterals. Each quadrangular cell of the Lagrangian mesh is delimited by the four fixed markers on its vertices. Grid intersection and area conservation markers are positioned on the segment connecting two consecutive fixed markers at the beginning of the first time step, with the procedure discussed in Section 3.2. For all tests, we consider the fractional time step $\delta t = \mathrm{CFL}/10$ where the CFL number is the non-dimensional velocity $\mathrm{CFL} = u^*\Delta t/h$, where u^* is the maximum of the velocity components in the computational domain, Δt the time step and h the grid spacing. In 2D the interface line lies on a plane, resulting in a more relaxed fractional step, $\delta t = \mathrm{CFL}/4$. Experimentally, this δt appears to be a good compromise between computational efficiency and accuracy of both volume and geometric errors that decrease as the markers are advected more accurately along the streamlines. We remark that lower order integration schemes have been considered as well, but they require a considerably lower value for the fractional step. In a direct numerical simulation of multiphase flows it is usually $\mathrm{CFL} \ll 0.1$, because of numerical stability constraints, and the integration scheme can be applied only once. Furthermore, it is also simple to adapt the number of integrations to the local CFL number.

5.3.1. Translation

We place a fluid body in the center of a unit box that is partitioned by cubic cells of constant grid spacing h along the three coordinate directions. In particular, we take $n = 1/h = 32$ and a uniform and constant in time velocity field $\vec{u} = (u, v, w)$ with equal velocity components so that both phases translate diagonally across the mesh. Cylindrical, spherical and conical fluid bodies have been tested. The computations have been performed with different CFL numbers always resulting in zero volume and geometrical errors, since the streamlines are always computed exactly.

5.3.2. Rotation

We consider the same unit cube and $n = 1/h = 32$. A constant-vorticity velocity field is imposed with the rotation axis parallel to the x-axis. The velocity field is

$$\begin{aligned} u(x,y,z) &= 0, \\ v(x,y,z) &= -\Omega(z - z_c), \\ w(x,y,z) &= \Omega(y - y_c), \end{aligned} \tag{19}$$

where $\Omega = 2\pi/200$ is the angular velocity and $(0, y_c, z_c)$ are the coordinates of the intersection point of the rotation axis with the plane $x = 0$. For this test CFL $= 32\pi/200$ and we take $\delta t =$ CFL$/10$. This solenoidal field rotates a fluid body in the counterclockwise direction for a viewer positioned at the origin of the coordinates system. In Fig. 16 we plot the initial discretized surface of a sphere with radius $r = n/8$ centered at $(n/2, n/2, 5n/16)$ and at times $t = T/4$, $T/2$ and $3T/4$. In the same velocity field we then rotate a conical surface with the same radius and height $H = n/4$ centered at $(n/2, n/2, 6n/16)$. The results are shown in Fig. 17, each frame after a quarter of a period, as for the sphere. The streamlines are now curved but the

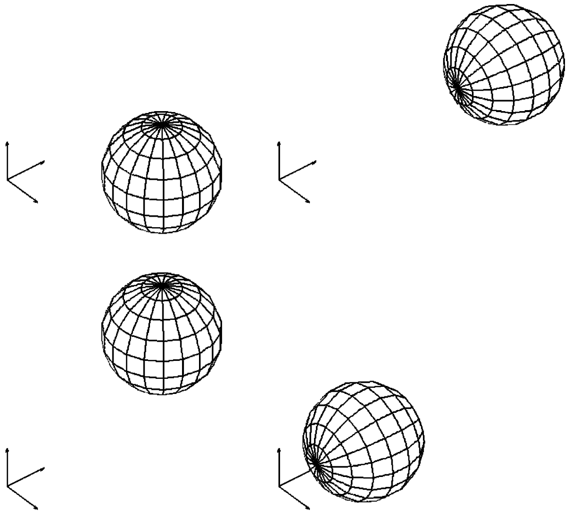

Fig. 16. Results for the sphere rotation on a 32^3 grid at times $t = 0, T/4, T/2, 3T/4$.

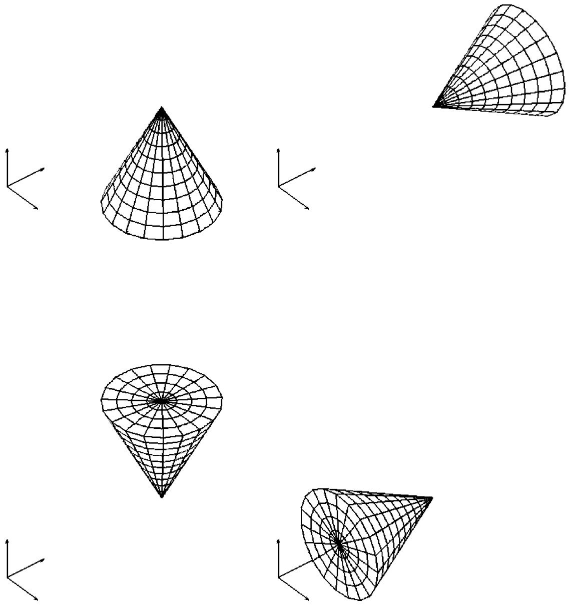

Fig. 17. Results for the cone rotation on a 32^3 grid at times $t = 0, T/4, T/2, 3T/4$.

body is not deformed, and for such a simple flow field our tracking algorithm has a geometrical error down to machine precision. The volume error is always exactly zero.

5.4. Three-dimensional deformation field test

We consider a three-dimensional incompressible flow field, similar to that proposed in [15], which is the superposition of three two-dimensional "vortex-in-a-box" fields, each of them deforming and stretching a fluid body on a different coordinate plane. The spatial velocity field is given by

$$\begin{aligned} u(x,y,z) &= \sin^2(\pi x)(\sin(2\pi z) - \sin(2\pi y)), \\ v(x,y,z) &= \sin^2(\pi y)(\sin(2\pi x) - \sin(2\pi z)), \\ w(x,y,z) &= \sin^2(\pi z)(\sin(2\pi y) - \sin(2\pi x)), \end{aligned} \tag{20}$$

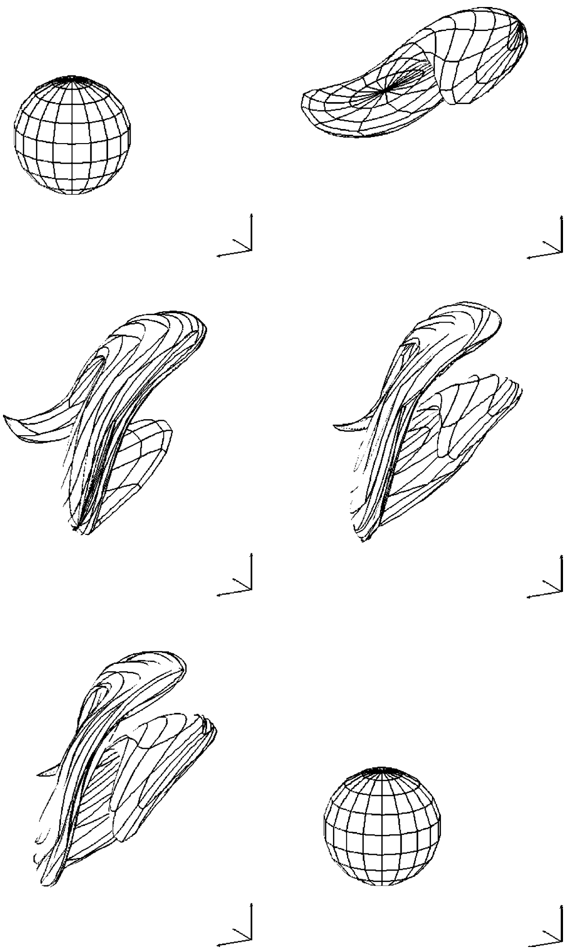

Fig. 18. Results for a sphere in the incompressible deformation field on a 32^3 grid with $T = 4$ at times $t = 0, T/8, T/4, 3T/8, T/2, T$, from left to right, top to bottom.

with the usual cosinusoidal time modulation with periods $T = 2$ and $T = 4$. The largest velocity is located half a way between the center of the box and the walls. A compact fluid body in this flow is deformed slantwise and progressively becomes a filament that is stretched and spirals around the box center. In Fig. 18 the reference phase is initially a fluid sphere with center at $(0.5, 0.75, 0.5)$ and radius $r = 0.15$. We plot the interface shape at different times with $T = 4$. In particular at $t = T/2$ the filament has completed two rotations around the center and at $t = T$ the fluid is back to its starting shape without any evident difference even for a rather coarse grid resolution with 32^3 cells. A more precise indicator of the algorithm accuracy in terms of volume and geometric errors, and not simply crude visual inspection, can be obtained by calculating the integrals in (14)–(16) with the initial and final phase indicator function. In Table 3 the errors are computed for the resolution 32^3, periods $T = 2$, 4, and CFL numbers equal to 1.0, 0.1, 0.01. In Table 4 we set both the period ($T = 2$) and the CFL number (CFL $= 0.1$) and consider several grid resolutions, namely $n = 16, 32, 64$. In all cases the volume and geometric errors decrease considerably with lower CFL numbers, i.e. smaller time steps, and higher grid resolution, i.e. smaller grid spacing. As a matter of fact, streamlines are better resolved at low CFL by the fourth-order Runge–Kutta algorithm and the discrete interpolated velocity field is closer to be divergence-free with grid refinement. The interface never breaks during all simulations and at the end of the period shows a stable and correct pattern even for the 16^3 mesh. At time $t = T/2$ the simulation with period $T = 4$ and $n = 32$ consists of $N_F + N_I + N_C = 400 + 2822 + 4526 = 7748$ markers while at $t = T$ the number of fixed markers N_F is the same, but there are considerably fewer intersection N_I and conservation N_C markers, $N_F + N_I + N_C = 400 + 666 + 1383 = 2449$. If we consider all cells near the interface within a distance of three grid cells, then we have 9318 and 3523 cells at $t = T/2$ and T, respectively. These two values have to be multiplied by the number of particles per cell in a particle method, usually a few tens.

We have also studied the sensitivity of the results to the initial number of lines. With reference to Fig. 7 we call *nv* the number of vertical lines and *nh* the number of horizontal lines. In Table 5 we compare the errors for a sphere in the 3D deformation field test with $n = 32$, $T = 2$ and three different couples of values $(nv, nh) = (5, 10), (10, 20), (20, 40)$, where 10 and 20 are the number of vertical and horizontal lines, respectively, used in all other simulations. If we consider that the number of fixed points changes by a factor

Table 3
Volume (E_v) and geometric (E_g^1, E_g^2) errors for a sphere in the single vortex test on a 32^3 grid with periods $T = 2$ and $T = 4$, at different CFL numbers

n	CFL	E_v	E_g^1	E_g^2
$T = 2$				
32	1.00	3.99e−3	9.40e−5	7.00e−3
32	0.10	6.10e−4	1.57e−5	1.17e−3
32	0.01	5.21e−5	7.25e−6	5.38e−4
$T = 4$				
32	1.00	7.83e−3	2.65e−4	1.99e−2
32	0.10	1.26e−3	4.50e−5	3.35e−3
32	0.01	1.82e−4	1.70e−5	1.26e−3

Table 4
Volume (E_v) and geometric (E_g^1,E_g^2) errors for a sphere in the single vortex test with $T = 2$ and CFL $= 0.10$, at different grid resolutions

n	CFL	E_v	E_g^1	E_g^2
16	0.10	4.83e−3	9.78e−5	7.30e−3
32	0.10	6.10e−4	1.57e−5	1.17e−3
64	0.10	6.50e−5	3.04e−6	2.26e−4

Table 5
Volume (E_v) and geometric (E_g^1,E_g^2) errors for a sphere in the single vortex test on a 32^3 grid with $T = 2$ and CFL = 0.10 at different numbers of vertical *nv* and horizontal *nh* interface lines

n	CFL	nv	nh	E_v	E_g^1	E_g^2
32	0.10	5	10	6.41e−4	1.96e−5	1.77e−3
32	0.10	10	20	6.10e−4	1.57e−5	1.17e−3
32	0.10	20	40	6.54e−4	1.39e−5	9.93e−4

of four by doubling the number of vertical and horizontal lines, we conclude that the mass and geometric errors are relatively insensitive to the total number of lines used to reconstruct the interface.

In Figs. 19 and 20 we show the deformation of a cylinder and a cone both with radius $n/8$, height $n/4$ centered at $(0.5, 0.75, 0.5)$ in the same velocity field (20) but with period $T = 2$. We point out that the cylinder edges and the cone vertex are very well recovered at the end of the simulation because the fixed points lying on these edges are advected with great accuracy. The performance of the algorithm in terms of volume and geometric errors is reported in Tables 6 and 7. The technique shows a very good global mass conservation and further investigations should be done to confirm these results in different dynamical situations. Furthermore, we have already shown in Fig. 1 the ability of the method to deal with several different objects in the same deforming field with period $T = 4$. The results are very similar with either 32 or 64 cells along each coordinate direction.

Finally, we remark that the method can be extended to 3D compressible flows. Here, we consider only the kinematics aspects of interface tracking, while a complete assessment of the mass conservation property would require the solution of the coupled system including the phase indicator advection equation and the continuity equation. Furthermore, we restrict the presentation to continuous velocity fields, where the interpolation technique described in the previous sections can be directly applied. We position the cylinder of Fig. 19 in the center of the unit box, with the spatial part of the velocity field defined as

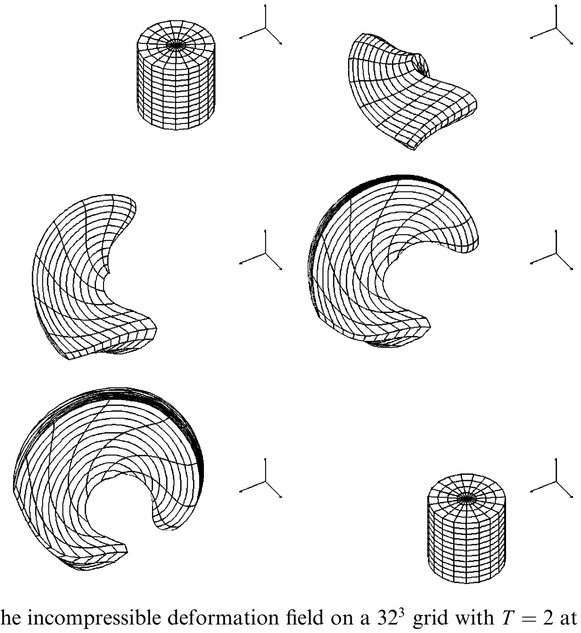

Fig. 19. Results for a cylinder in the incompressible deformation field on a 32^3 grid with $T = 2$ at times $t = 0, T/8, T/4, 3T/8, T/2, T$, from left to right, top to bottom.

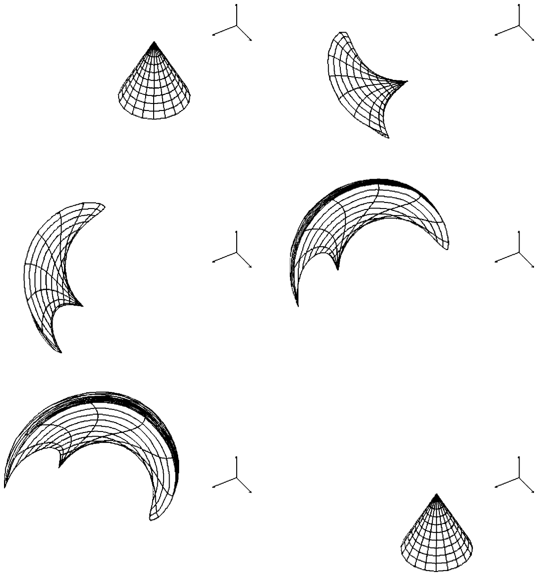

Fig. 20. Results for a cone in the incompressible deformation field on a 32^3 grid with $T = 2$ at times $t = 0, T/8, T/4, 3T/8, T/2, T$, from left to right, top to bottom.

Table 6
Volume (E_v) and geometric (E_g^1,E_g^2) errors for a cylinder and a cone in the single vortex test on a 32^3 grid with period $T = 2$, at different CFL numbers

n	CFL	E_v	E_g^1	E_g^2
Cylinder				
32	1.00	4.17e−3	8.76e−4	7.30e−3
32	0.10	6.54e−4	1.53e−5	1.27e−3
32	0.01	7.78e−5	5.60e−6	4.65e−4
Cone				
32	1.00	2.90e−3	5.58e−5	1.40e−2
32	0.10	4.45e−4	1.00e−5	2.50e−3
32	0.01	4.04e−5	2.40e−6	5.98e−4

Table 7
Volume (E_v) and geometric (E_g^1,E_g^2) errors for a cylinder and a cone in the single vortex test with $T = 2$ and CFL = 0.10, at different grid resolutions

n	CFL	E_v	E_g^1	E_g^2
Cylinder				
16	0.10	1.98e−3	9.94e−5	8.27e−3
32	0.10	6.54e−4	1.53e−5	1.27e−3
64	0.10	6.25e−5	2.45e−6	2.03e−4
Cone				
16	0.10	4.05e−3	7.14e−5	1.79e−2
32	0.10	4.45e−4	1.00e−5	2.50e−3
64	0.10	4.19e−5	1.62e−6	4.06e−4

$$u = 0.5 - x, \quad v = 0.5 - y, \quad w = 0.5 - z, \tag{21}$$

and a cosinusoidal time dependence. In this flow, all velocity vectors point to the box center and a fluid body first shrinks and then expands uniformly with no deformation. We can follow the time evolution of the cylinder back to its initial position with the volume and geometric errors always equal to zero, since the streamlines are straight and are traced exactly. In a more difficult test, we sum the compressional field (21) with the deforming flow (20). The cylinder is now initially positioned as in Fig. 19 and the interface evolution is shown in Fig. 21. The effects of compression can be appreciated by comparing these two figures. To be more quantitative, the volume and geometric errors, presented in Table 8 for the compressible deforming flow as a function of both CFL number and grid resolution, should be compared with the results of Tables 6, 7. In the considered ranges of time steps and grid spacings the errors for the two different flows are rather similar.

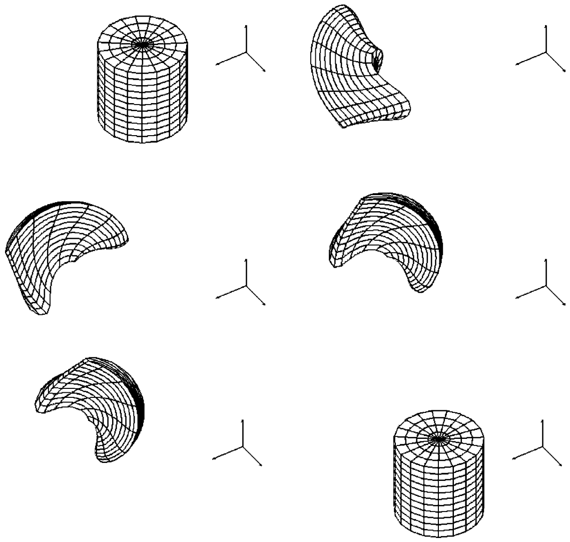

Fig. 21. Results for a cylinder in the compressible deformation field on a 32^3 grid with $T = 2$ at times $t = 0, T/8, T/4, 3T/8, T/2, T$, from left to right, top to bottom.

Table 8
Volume (E_v) and geometric (E_g^1,E_g^2) errors for a cylinder in the compressible single vortex test with $T = 2$, at different CFL numbers and grid resolutions

n	CFL	E_v	E_g^1	E_g^2
CFL numbers				
32	1.00	4.33e−3	7.39e−5	6.16e−3
32	0.10	8.39e−4	1.54e−5	1.28e−3
32	0.01	1.31e−4	6.03e−6	5.00e−4
Grid resolutions				
16	0.10	6.10e−3	9.76e−5	8.17e−3
32	0.10	8.39e−4	1.54e−5	1.28e−3
64	0.10	1.07e−4	3.38e−6	2.81e−4

6. Conclusion

A new coupled marker and local area conservation method for interface tracking in both the two-dimensional and three-dimensional spaces has been presented. The 2D algorithm greatly improves the performance of the first version of this method originally called a "mixed markers and VOF" method. The local area conservation methodology as opposed to the conservation of the volume fraction C in any given cell, greatly simplifies the algorithm which is at the same time much faster and more accurate than its previous version. Results are far superior than any other approach based on a single scalar function, such as the volume fraction C or the level set function φ, and compare favorably with those obtained with other mixed techniques such as the hybrid particle level set method. The method has been extended to the three-dimensional space where the interface is represented by a set of closed lines. Fixed markers are located at the intersection points of these lines and are kept during the whole simulation, while grid intersection and area conservation markers are added and removed dynamically where needed. Lines are moved separately and independently and the overall method remains two-dimensional in the 3D space as well. Tests performed with deforming incompressible and compressible flows have shown the ability of the technique to follow with great accuracy stretching interfaces in regions of very high curvature and where they develop long filamentary structures while other methods have to resort to much smaller grid spacing or to adaptive mesh refinement together with some degree of smoothing.

References

[1] R. Adams, Sobolev Spaces, Academic Press, New York, 1975.

[2] E. Aulisa, S. Manservisi, R. Scardovelli, A mixed markers and volume-of-fluid method for the reconstruction and advection of interfaces in two-phase and free-boundary flows, J. Comput. Phys. 188 (2003) 611.

[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] S. Chen, D. Johnson, P. Raad, D. Fadda, The surfer marker and micro cell method, Int. J. Numer. Meth. Fluids 25 (1997) 749.

[5] D. Enright, R. Fedkiw, J. Ferziger, I. Mitchell, A hybrid particle level set method for improved interface capturing, J. Comput. Phys. 183 (2002) 83.

[6] 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.

[7] J. Glimm, J.W. Grove, X.L. Li, K.-M. Shyue, Y. Zeng, Q. Zhang, Three-dimensional front tracking, SIAM J. Sci. Comput. 19 (1998) 703.

[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] S. Guignard, R. Marcer, P. Fraunie, Solitary wave breaking on sloping beaches: 3-D two-phase flow simulation by SL-VOF method, Eur. J. Mech. B – Fluids 20 (2001) 57–74.

[10] 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.

[11] D.J.E. Harvie, D.F. Fletcher, A new volume of fluid advection algorithm: the stream scheme, J. Comput. Phys. 162 (2000) 1.

[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] M. Kang, R. Fedkiw, X.D. Liu, A boundary condition capturing method for multiphase incompressible flow, SIAM J. Sci. Comput. 15 (2000) 323–360.

[15] R.J. Leveque, High-resolution conservative algorithms for advection in incompressible flow, SIAM J. Numer. Anal. 33 (1996) 627.

[16] S. Mosso, B. Swartz, D. Kothe, R. Ferrell, A parallel, volume-tracking algorithm for unstructured meshes, in: P. Schiano, A. Ecer, J. Periaux, N. Satofuka (Eds.), Parallel Computational Flui Dynamics: Algorithms and Results using Advanced Computers, Elsevier Science, Capri, Italy, 1997, pp. 368–375.

[17] S. Osher, R. Fedkiw, Level Set Methods: an overview and some recent results, J. Comput. Phys. 169 (2001) 463–502.

[18] S. Popinet, S. Zaleski, A front-tracking algorithm for accurate representation of surface tension, Int. J. Numer. Meth. Fluids 30 (1999) 775.

[19] 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.

[20] 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.

[21] W.J. Rider, D.B. Kothe, Stretching and tearing interface tracking methods, in: 12th AIAA CFD Conference, AIAA, 1995, p. 95.

[22] W.J. Rider, D.B. Kothe, Reconstructing volume tracking, J. Comput. Phys. 141 (1998) 112.

[23] R. Scardovelli, S. Zaleski, Direct numerical simulation of free-surface and interfacial flow, Ann. Rev. Fluid Mech. 31 (1999) 567.

[24] R. Scardovelli, S. Zaleski, Interface reconstruction with least-square fit and split Eulerian–Lagrangian advection, Int. J. Numer. Meth. Fluids 41 (2003) 251.

[25] J. Sethian, Evolution, implementation and application of level set and fast marching methods for advancing fronts, J. Comput. Phys. 169 (2001) 463.

[26] 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.

[27] P.K. Smolarkiewicz, The multi-dimensional Crowley advection scheme, Monthly Weather Rev. 110 (1982) 1968.

[28] G. Son, V.K. Dhir, Numerical simulation of film boiling near critical pressures with a level set method, J. Heat Trans. 120 (1998) 183.

[29] M. Sussman, A. Almgren, J. Bell, P. Colella, L. Howell, M. Welcome, An adaptive level set approach for incompressible two-phase flows, J. Comput. Phys. 148 (1999) 81.

[30] M. Sussman, E. Fatemi, P. Smereka, S. Osher, A level set approach for computing solutions to incompressible two-phase flow, J. Comput. Phys. 114 (1994) 146.

[31] 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.

[32] D. Torres, J.U. Brackbill, The point-set method: front tracking without connectivity, J. Comput. Phys. 165 (2000) 620.

[33] 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.

[34] S.O. Unverdi, G. Tryggvason, A front-tracking method for viscous, incompressible, multi-fluid flows, J. Comput. Phys. 100 (1992) 25.

[35] S. Welch, J. Wilson, A volume of fluid based method for fluid flows with phase change, J. Comput. Phys. 160 (2000) 662.