一つの計算格子で一つの気泡を捕らえる高精度ＶＯＦ法

A one-grid-one-bubble high resolution VOF method

○　孫　明宇, 東北大学際センター, 仙台市青葉区荒巻字青葉, E-mail: sun@cir.tohoku.ac.jp
Mingyu Sun, Center for Interdisciplinary Research, Tohoku University, Sendai 980-8578, Japan

A new PLIC-VOF method is proposed to improve the accuracy of VOF method in resolving fluid structures at low grid resolution. The idea is to replace the surface normal calculation used in the PLIC, by using the surface normal (SN) vector given by partial differential equations that are solved together with the advection algorithm. The equations for the evolution of surface normal vector are derived, and examined by using the first-order and the second-order upwind schemes. It is found that the normal vector with zero magnitude is located at the centroid of a subgrid particle, to second-order accuracy. The motion of a subgrid particle is controlled by the arrival of the normal vector with zero magnitude in the present PLIC/SN method, so that the particle can be translated at the right speed without any additional treatment. This has been numerically verified by simulating a particle of one tenth of grid spacing traveling in three different directions, in addition to a few typical test cases.

1. Introduction

The volume-of-fluid (VOF) is one of the most widely used methods for the numerical simulation of interfacial phenomena. In the VOF method, the evolution of an interface is predicted generally by solving the following advection equation

$$\phi_t + \mathbf{u} \cdot \nabla \phi = 0, \tag{1}$$

where $\mathbf{u}$ is the velocity. Function ϕ is the volume fraction or color function at the discrete level. It is unity in a cell filled with one phase, and is zero if the cell lies in the other phase. Excellent reviews on this subject have been given by Rider and Kothe (1998) and Scardovelli and Zaleski (1999) so only brief review of the related methods will be given here. The majority of VOF methods are based on two key procedures. One is to reconstruct an interface in a cell, and the other to advance the volume fraction in time and space. Both are not trivial, because the volume fraction is discontinuous across a sharply resolved interface. The VOF methods are different mainly at the way to handle these two procedures.

In VOF, the interface in a cell is not tracked, but reconstructed and approximated by a simple geometry. The simple line (piecewise constant) interface calculation (SLIC) assumes the geometry is a line parallel to one of the grid lines. Currently most widely used method is based on the piecewise linear interface calculation (PLIC). In PLIC, the surface normal vector, $\mathbf{n}$, is required to construct the linear interface

$$\mathbf{n} \cdot \mathbf{r} + h = 0, \tag{2}$$

where h is the line constant. The normal vector is inferred from the spatial distribution of ϕ. If ϕ is a smooth function, the normal vector satisfies

$$\mathbf{n} = \nabla \phi. \tag{3}$$

A few numerical methods have been proposed for the calculation of the surface normal. Youngs (1984) used the finite-difference method to discretize the gradient in (3) directly. Puckett (1991) approximated the surface normal vector from the volume fraction in a 3×3 block of cells using an iterative method. The efficiency of the iterative method was improved by Pilliod using an algorithm, ELVIRA, which can reconstruct all linear interfaces exactly. The reconstruction using a spline interface was attempted by López et al. (2004). The volume fraction is advanced from the geometry of the reconstructed interface, by multidimensional/unsplit schemes or one-dimensional/operator-split schemes. Continuous efforts have been made to improve the advection algorithms.

It is quite well known that the VOF method will be accurate whenever the radius of curvature is large with respect to the grid size. It will be less accurate for scales comparable, and may lose all details for scales smaller than one grid spacing. Consider fluid structures with the characteristic length, d, say, the thickness of a thin filament. If d reduces to the order of the grid size, Δx, the VOF method numerically splits or merges them. It is called numerical surface tension by Rider and Kothe (1998). Černe et al. (2002) investigated the behavior of the VOF method in simulating these small interface structures, and found that the error of the interface reconstruction increases rapidly when $d/\Delta x \leq 3$. This is in agreement with the fact that a minimum of three grid cells are required to resolve a circular particle with certain accuracy. They also found that the advection errors occur as well. The small particle moves faster for $d/\Delta x < 2$. Subgrid particles ($d/\Delta x < 1$) were not considered in their study.

For resolving subgrid particles, there are two difficulties behind the PLIC-VOF method. Consider an isolated subgrid particle in a cell, as illustrated in Fig. 1a. The subgrid particle generally stays in the cell, surrounded by the dark phase in neighboring cells. The first difficulty is that the gradient of volume fraction (3) cannot provide a meaningful solution. How to update the volume fraction is another difficulty. In order to enhance the accuracy of VOF at low grid resolution, more information on the interface in addition to the volume fraction is necessary. López et al. attempted to resolve thin filaments by using makers (2005).

This work tries to improve the accuracy of the PLIC-VOF method at low grid resolution, and to expand its capability in dealing with subgrid particles. The idea is to treat the surface normal vector as independent variables that are integrated along with the advection algorithm, instead of calculating them from (3).

For the purpose of resolving subgrid interfaces, the equations for the surface normal vector are derived without knowing the work of Raessi et al. (2007), who considered the same equations imposed with unit normal constraint $|\mathbf{n}| = 1$, and obtained accurate surface curvatures for resolvable circular interfaces. In this work, the treatment for surface normals has been developed independently and treated in a very different way. We came up with the same idea of using the unit normal at the early stage of development, but soon it was found that the unit normal creates a singularity at the center

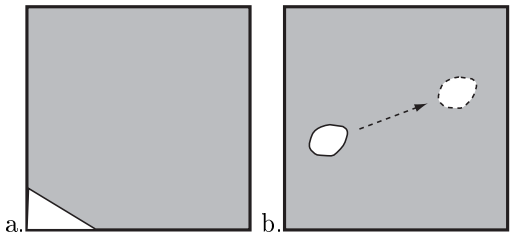

Fig. 1: A PLIC-VOF method may reconstruct a subgrid particle, given the phase volume and the normal vector. However, the normal vector cannot be properly calculated from (3) only from the distribution of volume fraction for the isolated particle. Even a vector is assigned as shown in (a) by certain rule based on the volume fraction alone, the particle will be trapped in the cell. Consider a moving particle as shown in (b). Although the particle has traveled a short distance within the cell, the PLIC-VOF method will reconstruct the same linear segment in the cell because the volume fraction and thus its distribution are unchanged.

of a circle, where the surface normal is discontinuous after normalization. The numerical error originating from the singularity is a disaster that can pollute the solution far from the center, and thus the method is not good for tracking small particles. This is in agreement with the observation that numerical errors for small particles are large as commented by Raessi et al. The problem was solved by using a smooth surface normal distribution with its magnitude decreasing to zero at the center, which is not the unit vector. Another problem was encountered when dealing with an interface with sharp corners (e.g. a square interface, to be shown in section 5) and for compressible flows with discontinuous velocity field. To solve the problem, the non-conservative equations for surface normals are proposed and generally followed in our work (section 2). In short, the present work is different from their work at (1) the non-conservative equations for surface normals are proposed and generally tested, and (2) unit normal constraint is not imposed.

This paper is organized as follows. The advection equations for surface normal vector are derived in section 2. The relationship between the surface normal and the motion of a particle is shown in section 3. All issues on numerical discretization are documented in section 4. Both the first and the second order upwind schemes are implemented. Their results are compared and discussed in section 5. The behavior of a subgrid particle motion is investigated in section 5.4.

2. Equations for the evolution of surface normal vector

Suppose function ϕ is smooth enough, the gradient of (1) leads to,

$$(\nabla\phi)_t + \nabla(\mathbf{u}\cdot\nabla\phi) = 0.$$

Replacing $\nabla\phi$ by normal vector $\mathbf{n}$ as defined in (3), one gets

$$\mathbf{n}_t + \nabla(\mathbf{u}\cdot\mathbf{n}) = 0, \tag{4}$$

which represents the evolution of the surface normal vector. Unlike the original advection equation (1), the surface normal equation is conservative for any velocity field. The flux function is just the inner product of velocity and surface normal vector. For two dimensional problems, the surface normal equation can be expressed explicitly as, letting $\mathbf{n} = (l, m)$,

$$\begin{aligned} l_t &+(ul+vm)_x = 0, \\ m_t &+(ul+vm)_y = 0. \end{aligned} \tag{5}$$

For a smooth scaler field ϕ satisfying $\phi_{xy} = \phi_{yx}$, two components of the surface normal vector then satisfy

$$l_y = m_x. \tag{6}$$

The derived equation (5) does not contain the original function ϕ, which allows us to treat the surface normals as two independent variables.

For 1-D problems, the equations degenerate to

$$l_t \quad +(ul)_x = 0, \tag{7}$$

which is identical to the mass equation in the Euler equations, if replacing l by density. Its mathematical and numerical behavior is therefore similar to the density in 1-D gas-dynamic problems.

For 2-D problems, it is of interest to see that every equation in (5) contains only the spatial derivative in its own direction, which is in general different from the mass equation. Numerically, it implies that a numerical scheme for (5) is lack of any stabilizing mechanism for numerical disturbances possibly appearing in the other direction. Using (6), equations (5) can be rewritten as

$$\begin{aligned} l_t &+(ul)_x + (vl)_y &&= -mv_x + lv_y, \\ m_t &+(um)_x + (vm)_y &&= mu_x - lu_y, \end{aligned} \tag{8}$$

in which the left-hand side (LHS) of the equations is the same as the 2-D mass equation. The source terms on the right-hand side represent the velocity gradient along an interface, or the rotation of the surface by the velocity field. Notice that for a constant velocity field, both equations in (8) degenerate to the 2-D linear wave equation, but (5) does not. Both sets of equations are to be investigated and compared numerically.

In this work, two surface normal components are solved as they are, without normalizing them during the course of integrating the equations, although the normalization of a surface normal vector does not change the orientation and the curvature of an interface. The main reason is that the normalization creates one or more discontinuities inside the particle, for example at the center of a circle, thus reduces the numerical accuracy there. A continuous normal vector is also required to maintain a small particle moving at the right flow velocity (section 3.).

3. The surface normal and the motion of a small particle

We shall consider the mathematical relationship between the surface normal and the motion of a particle in this section. Let domain D occupied by a resolvable or subgrid particle be a simply connected domain bounded by the surface $\Gamma : \phi = c$. The following properties will be proved for the surface normal of the scalar field ϕ that is smooth enough to define $\mathbf{n} = \nabla\phi$.

Lemma: The surface normal in the domain D satisfies

$$\int_D \mathbf{n} dA = 0. \tag{9}$$

Proof: Consider the following integral,

$$\int_D \mathbf{n} dA = \int_D (\nabla\phi) dA,$$

one has, using the Stokes' theorem for x-component,

$$\int_D (\frac{\partial \phi}{\partial x}) dA = \oint_\Gamma \phi dy = 0,$$

for $\phi = c$. Other components can be proved to be zero in the similar way.

This lemma states that no matter how the surface deforms, the total normal vector in the particle is zero. In order to characterize the size of a deformable particle, we introduce length scale, d, such that the surface of the particle can be bounded by a circle with the minimum diameter d.

Theorem 1: The surface normal at the centroid of a particle, $\mathbf{n}_c$, is the zero vector within the error of $O(d^2)$,

$$\mathbf{n}_c = \mathbf{0} + O(d^2).$$

Proof: Approximating $\mathbf{n}$ by the Taylor series expansion at the centroid gives

$$\int_D \mathbf{n} dA = \int_D [\mathbf{n}_c + (\nabla \mathbf{n}) \cdot (\mathbf{r} - \mathbf{r}_c) + O(d^2)] dA = 0,$$

where $O(d^2)$ represents omitted second and higher order terms. Using the definition for the centroid, $\int_D (\nabla \mathbf{n}) \cdot (\mathbf{r} - \mathbf{r}_c) dA = 0$, gives

$$\int_D [\mathbf{n}_c + O(d^2)] dA = 0.$$

This completes the proof.

This theorem states that whatever the initial shape of the particle is, if it is shrunk to be very small $d \to 0$, the surface normal vector at its mass center will be reduced to the zero vector as well. In the other word, the zero surface normal represents the location of the mass center of the particle within the error of $O(d^2)$.

If velocity $\mathbf{u}$ is that of fluid particles, we have

Theorem 2: The zero surface normal moves at the particle velocity of $\mathbf{u}$.

Proof: Consider a circular fluid particle with its centroid moving at the speed of $\mathbf{u}$. Since the zero surface normal is located at the centroid within the error of $O(d^2)$, the zero vector moves at a speed of $\mathbf{u} + O(d^2)$, and then the speed converges to $\mathbf{u}$ for the infinitely small particle ($d \to 0$).

A surface normal other than the zero vector generally does not move at the speed of $\mathbf{u}$. In addition to the translational motion, the surface normal vector undergoes rotation, compression or expansion, which actually represents the deformation of surfaces. This theorem can be seen from the surface normal equations as well. In the neighborhood of the zero vector, for $l \to 0$ and $m \to 0$, equations in (8) become, by omitting terms with l and m,

$$\begin{aligned} l_t &+ u l_x + v l_y &= 0, \\ m_t &+ u m_x + v m_y &= 0, \end{aligned} \tag{10}$$

which show that both surface normals move at the speed of $\mathbf{u} = (u, v)$. It is emphasized that the conservative equations in (5), however, lead to

$$\begin{aligned} l_t &+ u l_x + v m_x &= 0, \\ m_t &+ u l_y + v m_y &= 0. \end{aligned} \tag{11}$$

Although they become the same if relation (6) is satisfied, it would be very tough to numerically integrate equations (5) while maintaining (6). The idea of using the non-conservative equations simply avoids this difficulty.

In short, for a small particle of $O(\Delta x)$, *the centroid of the particle moves at the same velocity as the zero surface normal, to second-order accuracy.* Whatever the size and shape of a subgrid particle is, this principle allows us to implicitly transport it by evolving the equation of surface normal (8), only if the particle follows the zero vector. The zero vector is a 'marker', not tracked but evolved by the partial differential equations.

4. The PLIC/SN method

Coupling the VOF method with the surface normal equations, the present volume tracking method is divided to three procedures,

(a) to reconstruct the piecewise linear interface with a given surface normal in each interface cell such that the interface truncates the cell with a fractional volume equaling the given phase volume in the cell (section 4.1);

(b) to advance the phase volumes using a unsplit algorithm with a limiter ensuring that all phase volumes lie within bounds of $[0, \Omega]$ (section 4.2);

(c) to advance the surface normal vectors (section 4.3).

Procedures (a) and (b) are nothing but a typical PLIC-VOF method except replacing the calculation of surface normal by using the surface normal vector obtained in (c). The surface normal vector is integrated by the finite volume method, to be discussed in section 4.3. Since the method combines the PLIC method with the surface normal (SN) equations, we shall refer to the present method as PLIC/SN in what follows.

The discretization procedures to be discussed are so optimized and formulated that they can be readily implemented on any grid system and coupled with a finite-volume flow solver. The only input required from the flow solver is the velocity field. All formulas and discussions are valid for structured and unstructured grids, except for the interface reconstruction procedure that is valid only for regular rectangular cells. The whole algorithm has been developed on a solution-adaptive unstructured quadrilateral grid by Sun & Takayama (1999), and coupled with a compressible flow solver. However, only the results on the Cartesian grid with specified velocity fields are reported in this paper for the purpose of comparison and evaluation.

4.1 Representation of resolvable and subgrid particles

The present method to resolve different sized particles is illustrated in Fig.2, which contains a circular particle with diameter d varied from $1/2\Delta x$ to $4\Delta x$. The interface representation is the same as what has been developed in the PLIC-VOF methods. The only difference is that the surface normal vector has been defined and updated by solving the equations, so that surface normal calculations are not necessary. With the surface normal vector defined in the cell, any sized particles can be unambiguously defined.

An interface is discretized by linear segments in the cells that intersect with the interface. Each linear segment, starting from and ending at the cell edges, divides the interface cell to two portions having the exact volume of two phases. Two linear segments defined in neighboring cells are in general not connected as shown

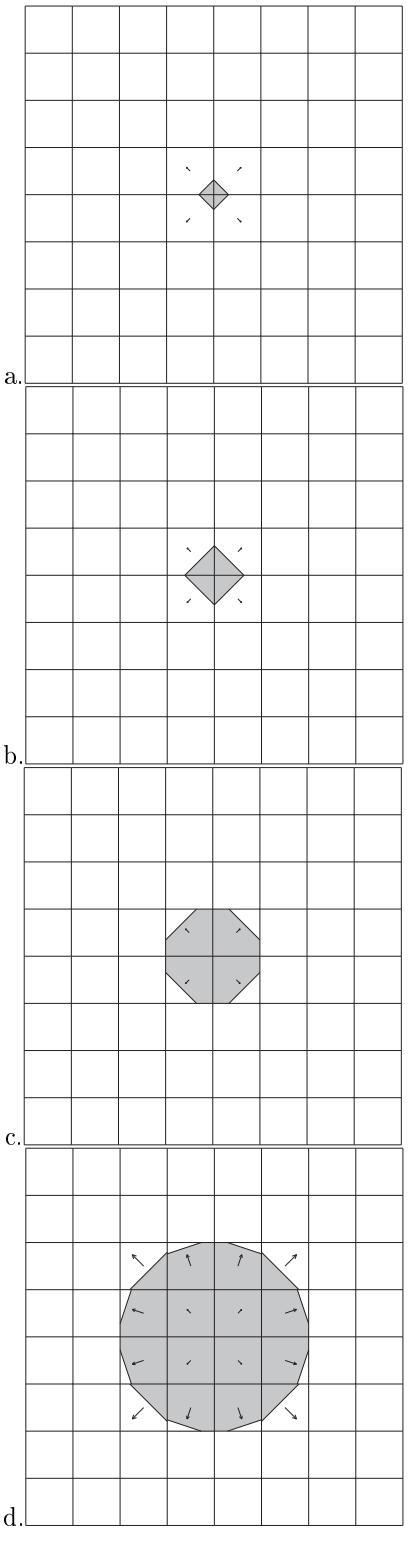

Fig. 2: Representation of different sized circular particles: a) $d/\Delta x = 1/2$; b) $d/\Delta x = 1$; c) $d/\Delta x = 2$; $d/\Delta x = 4$.

in Fig.2d, so that the segments themselves are not sufficient to shape a closed interface for a particle. The dark particle is actually bounded by the segments inside the cells together with the wet portion of edges between cells. The center of the subgrid circle shown in Fig.2a is located at the grid node, so that the subgrid circle is represented as a symmetric diamond. Its shape changes with the location of its center (see more examples in section 5, Fig. 4).

Given the surface normal vector $\mathbf{n}$ and the phase volume, the piecewise linear interface (2) is unique in a cell. In general, the line constant h needs to be found by iterative method, because the truncation volume is often a nonlinear function of h, especially for axisymmetric and 3-D geometries. For 2-D rectangular cells, the non-iterative method is followed to find h.

In short, in order to represent subgrid as well as large particles for any grid system in a simple way, what we follow is no more than

(1) *at most one linear material interface is located in a control volume;*

(2) *more material interfaces are allowed to be aligned with grid lines or faces between two neighboring control volumes, which together with the interface inside may form an isolated subgrid particle.*

No additional restrictions such as the size of a particle, the distance between particles are imposed in the present interface representation. Based the simple rules, the topological changes of a particle can be implicitly handled, as most VOF methods. If a particle is divided into two portions with a distance between greater than one grid spacing, it is regarded as the particle breakup, or if the sides of two particles meet in the same cell, they are merged automatically. For resolving two immiscible particles, the distance between them should be large enough to avoid numerical coalescence.

For rule (ii), no special treatment is really needed for a PLIC algorithm. The material interface aligned with grid lines appears as a natural result of PLIC. One example is seen in Fig. 4e. It is listed here because this sort of subgrid particles, which are unable to be properly handled in a traditional PLIC/VOF method, might have been merged or deleted.

4.2 Advection of phase volumes

The advection equation (1) is solved by the finite volume method. The volume flux is evaluated in the direction normal to the grid line, and integrated in a unsplit manner. Given a flow field $\mathbf{u}$, the advection equation is rewritten as

$$\phi_t + \nabla \cdot (\mathbf{u}\phi) = \phi \nabla \cdot \mathbf{u}, \tag{12}$$

and

$$\phi_t + \nabla \cdot (\mathbf{u}\phi) = 0, \tag{13}$$

for imcompressiable velocity field. Equation (13) is discretized by using the finite volume method,

$$\Omega_k^{n+1} = \Omega_k^n - \sum_j (u_n \phi_k \Delta t)_j, \tag{14}$$

where u_n is the outward normal velocity across grid interface j, and ϕ_k is the volume fraction of phase k defined at the grid interface, satisfying $\sum_k \phi_k = 1$. Ω_k is the volume of phase k, $\Omega_k = \phi_k \Omega$, satisfying

$$\Omega = \sum_k \Omega_k. \tag{15}$$

Without losing generality, the phase with the smaller volume is denoted as *slave* volume, Ω_s, and the other phase *master* volume, Ω_m. In the present method, the slave volume is integrated by (14),

$$\Omega_s^{n+1} = \Omega_s^n - \sum_j \Phi(u_n \phi_s \Delta t)_j, \tag{16}$$

where Φ is a limiter function to ensure

$$\Omega_s^{n+1} \in [0, \Omega]. \tag{17}$$

The limiter function is defined such that the outflow of the slave volume should be no more than that inside,

$$\Phi = \begin{cases} \frac{\Omega_s^n}{f_s} & \text{for} \quad f_s > \Omega_s^n \\ 1 & \text{otherwise,} \end{cases} \tag{18}$$

where f_s is total outflow volume flux of the slave volume,

$$f_s = \sum_{j, u_n > 0} (u_n \phi_s \Delta t)_j.$$

The master volume is updated by using (15),

$$\Omega_m^{n+1} = \Omega - \Omega_s^{n+1}. \tag{19}$$

Clearly, if the slave volume satisfies (17), then Ω_m^{n+1} lies within the range as well. It is readily seen that Ω_s^{n+1} is always non-negative with the limiter function. If the volume change is below than $\Omega/2$, which is true for the CFL number below $1/2$, the slave volume starting from $\Omega_s^n \leq \Omega/2$ satisfies $\Omega_s^{n+1} \leq \Omega$. In short, the limiter function guarantees the exclusion of the overshoot and undershoot for both phases from numerical results.

It is noted that the Φ limiter is different from the redistribution algorithm adopted in some VOF algorithms. The redistribution algorithm is often triggered when an abnormal volume that is beyond $[0, \Omega]$ is found in the solution, and then redistribute the volume to somehow empirically chosen cells nearby. The Φ limiter is to adjust the volume flux such that the abnormal volume will not appear. The limiting method solves the problem beforewards, but the redistribution method does afterwards. The limiting method is general for any advection scheme used to evaluate volume fluxes.

Once the interface has been constructed, as discussed in section 4.1, the volume flux $(u_n \phi_s \Delta t)_j$ at grid line j is defined *geometrically* by calculating the volume or the area cut by the grid line shifted upstream by $-u_n \Delta t$, where u_n is the normal velocity defined in the upstream cell. When coupled with a compressible flow solver, the velocity can be defined by the Riemann solver. In this work, since the exact velocity field is specified in all test cases, u_n is simply taken as the exact solution located at the center of the grid line. The method to evaluate volume fluxes is 'naive unsplit'. A modification is made following DDR method proposed by Harvie and Flectcher (HA2001). Although the DDR method is less accurate than the advection algorithms that take into account the corner-coupled fluxing, we shall show that the accuracy obtained is comparable when it is coupled with the surface normal equations.

The minimum particle that can be resolved is restricted only by the machine accuracy in evaluating volumes and volume fluxes, and in the algorithm the slave volume satisfying

$$\Omega_s / \Omega \geq 10^{-13} N^{1/2} \tag{20}$$

is treated as an interface cell. N is the total cell number, and $N^{1/2}$ gives a simple estimate of the amplification of the round-off error due to two dimensional geometric calculations. The round-off error is set to be 10^{-13} for double precision floating-point computations.

4.3 Discretization of surface normal equations

Both conservative form (5) and non-conservative form (8) are implemented and investigated in this study. For the sake of clarity, they are rewritten as

$$\mathbf{n}_t + (\mathbf{F}_1)_x + (\mathbf{F}_2)_y = \mathbf{S}, \tag{21}$$

where for the conservative form (5)

$$\mathbf{F}_1 = \begin{pmatrix} ul + vm \\ 0 \end{pmatrix} \quad \mathbf{F}_2 = \begin{pmatrix} 0 \\ ul + vm \end{pmatrix} \quad \mathbf{S} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{22}$$

and for the non-conservative form (8)

$$\mathbf{F}_1 = \begin{pmatrix} ul \\ um \end{pmatrix} \quad \mathbf{F}_2 = \begin{pmatrix} vl \\ vm \end{pmatrix} \quad \mathbf{S} = \begin{pmatrix} -mv_x + lv_y \\ mu_x - lu_y \end{pmatrix}. \tag{23}$$

System (21) is discretized by the finite volume method. Consider a control volume Ω_i bounded by discrete faces with outward surface normal $\mathbf{s} = (s_x, s_y)$.

$$(\mathbf{n}_i)_t = \mathbf{S}_i - \frac{1}{\Omega_i} \sum_j \hat{\mathbf{F}}_j,$$

where the numerical flux $\hat{\mathbf{F}}_j$ is approximated by the upwind scheme, depending on the direction of normal velocity $u_n = \mathbf{u} \cdot \mathbf{s}$,

$$\hat{\mathbf{F}}_j = \begin{cases} \mathbf{F}_1^- s_x + \mathbf{F}_2^- s_y, & \text{if} \quad u_n > 0; \\ \mathbf{F}_1^+ s_x + \mathbf{F}_2^+ s_y, & \text{otherwise.} \end{cases} \tag{24}$$

Velocity and surface normal at faces are required to define the numerical fluxes.

4.3.1 First-order scheme For the first-order scheme in space, velocity and surface normal used to determine the numerical flux are simply those located in the control volume on the upwind side. The source terms are discretized using the central difference scheme.

4.3.2 Second-order scheme For achieving second-order accuracy in space, the surface normal and velocity at grid interfaces are located at the center of face, $\mathbf{r}_j^c$. They are interpolated from the center of the volume, following the MUSCL method,

$$M^{-,+} = M_i^{-,+} + (\nabla M)_i^{-,+} \cdot (\mathbf{r}^c - \mathbf{r}_i^{-,+}),$$

where M represents both the velocity and surface normal required to define the numerical flux. Superscripts $^{-,+}$ indicates the values are defined from left and right sides (or upstream and downstream) respectively. For example, $M_i^{-,+}$ are those values located at $\mathbf{r}_i^{-,+}$ respectively. In this work, they are defined at the cell center. $(\mathbf{r}_j^c - \mathbf{r}_i^{-,+})$ is the distance between the center of the grid interface and the location of values defined. It is simply a Taylor series expansion in multidimensions.

In solving the hyperbolic conservation laws, the limiter is often used to suppress possible numerical oscillations around discontinuities. To investigate the influence of the limiter on the solution of surface normal equations, the MINMOD slope limiter for limiting the gradient $(\nabla M)_i^{-,+}$ is also implemented. The source terms are discretized using the central difference scheme.

For achieving second-order accuracy in time, the two-step Runge-Kutta method is followed for the surface normal equations. However, the interface reconstruction and the volume flux evaluation, described in sections 4.1 and 4.2, are done only once based on the surface normal at the last time step.

4.3.3 Initial and boundary conditions For solving the IBV problems of surface normal equations, appropriate initial and boundary conditions must be specified for l and m. For a given interface, only the direction of surface normal vector is available nearby. Others have to be defined. An intuitive way to define these values is to construct a smooth surface function ϕ, such that $\phi = c$ represents the interface. Two types of particles, circle and square, are tested in this paper, and their definitions are summarized as below. For a circle centered at (x_0, y_0), we set

$$\phi(x,y) = \frac{1}{2}[(x-x_0)^2 + (y-y_0)^2],$$

and then differentiate it,

$$\begin{aligned} l(x,y) &= x - x_0, \\ m(x,y) &= y - y_0, \end{aligned} \tag{25}$$

which are the initial conditions for a circle. Similarly consider a square centered at (x_0, y_0) with sides parallel to grid lines, represented by

$$\phi(x,y) = \frac{1}{2}\begin{cases} (y-y_0)^2 & |y-y_0| > |x-x_0|; \\ (x-x_0)^2 & |y-y_0| < |x-x_0|; \\ \frac{1}{2}[(x-x_0)^2 + (y-y_0)^2] & |y-y_0| = |x-x_0|, \end{cases}$$

one gets two surface normals for squares,

$$l(x,y) = \begin{cases} 0 & |y-y_0| > |x-x_0| \\ x - x_0 & |y-y_0| < |x-x_0| \\ \frac{1}{2}(x-x_0) & |y-y_0| = |x-x_0|, \end{cases} \tag{26}$$

and

$$m(x,y) = \begin{cases} y - y_0 & |y-y_0| > |x-x_0| \\ 0 & |y-y_0| < |x-x_0| \\ \frac{1}{2}(y-y_0) & |y-y_0| = |x-x_0|. \end{cases} \tag{27}$$

Notice that the definitions are independent of the size of the circle or square, and it suggests the surface normals for any sized particles are solved equally.

Physical boundary conditions for surface normal vector of a real material interface attached on a wall depend on many factors, especially surface tension and gravity, which is beyond the scope of this paper. We shall focus on the interfaces inside the domain. In practice, for wall, inlet and outlet boundaries, we simply set

$$\frac{\partial l}{\partial s} = \frac{\partial m}{\partial s} = 0,$$

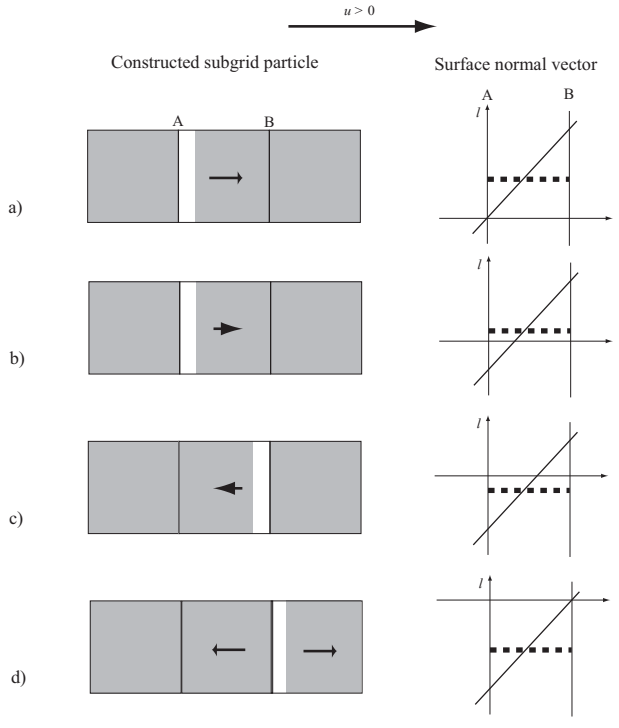

Fig. 3: A schematic of a 1-D subgrid particle motion controlled by the surface normal vector. The arrow always points from the bright phase to the dark phase in the left column, and its direction is determined by the sign of the averaged surface normal over the cell, the value of which is shown by the dashed line in the right.

where s is the direction normal to the boundary. This approximation is trivial for walls in real fluid flows, because both the velocity and the numerical fluxes are always zero there.

As an example, the trajectories for different sized moving particles are recorded and compared in Fig. 6. The diameter is varied from 0.1 to $12.8\Delta x$, covering the whole range from under-resolved to resolved scales. Two types of location error can be identified. One is the 'leap' error due to the interface reconstruction, which dominates in the motion of very small subgrid particles, as shown by the stepwise solid line with circles for $d/\Delta x = 0.1$. Another is the error resulting from the volume advection algorithm, as shown by the dashed line for $d/\Delta x = 12.8$. The advection error is much smaller and with a high frequency. For the particles between, these two errors are coupled. Nevertheless, it is clear that the motion of a subgrid particle has been properly resolved within the error of $O(\Delta x)$. Detailed formula and more numerical tests are shown in the full paper.

5. The advection of a subgrid particle

It has been shown that the zero normal vector moves at fluid velocity in section 3. We will investigate how a subgrid particle can follow the zero vector using the PLIC/SN method.

Consider a 1-D subgrid particle as shown in Fig. 3. Without losing generality, it is assumed that the surface normal points from the bright to the dark phase. Suppose the particle is initially located at face A, $x = x_0$. We use a linear function

$$l(x) = x - x_0$$

with zero exactly located at $x = x_0$ as the initial con-

dition for surface normal vector. The averaged surface normal over the cell bounded by faces A and B is then positive, as indicated by the dashed line in the top-right figure. Given the volume of the bright particle and the positive surface normal, the interface is then reconstructed on the left following the PLIC, as shown in Fig. 3a. Notice that the commonly used surface normal calculation for this small subgrid particle is senseless.

Now consider the motion of the particle. Suppose the flow moves at a constant speed to right, traveling through one cell in three steps. The evolution of the surface normal is shown in the right column. It starts with a positive value, and gradually reduces to be negative after the zero vector moves across the central point of the cell. After three steps, the zero vector has traveled from face A to face B, at the same speed as the flow. The corresponding reconstructed interface is shown in the left column. In the first step, from Fig. 3a to b, since the subgrid particle is attached on the face A, there is no volume flux through face B, thus the volume of the particle is unchanged. At this moment, the surface normal is still positive, so that the reconstructed interface is the same. Once the surface normal changes its sign, as seen from Fig. 3b to c, the particle is reconstructed on the right. The particle leaps from one side to the other side of the cell. In the third step, the particle is moved to the right neighboring cell by the advection algorithm, returning to the initial state but shifted by one grid cell. The procedure above will be repeated.

It is clear that the motion of a subgrid particle is made possible by both the leap controlled by the PLIC/SN and the volume advection through faces. The motion of a subgrid particle in two dimensions is more interesting and complicated. Fig. 4 shows all sequential steps of a particle of $d/\Delta x = 0.4$ moving at velocity $(u, v) = (1.0, 0.5)$. For this illustration, the CFL number is taken as $1/8$ based only on x-velocity, so that the particle is supposed to move from one node to the other in 16 steps, traveling two grid cells in x-direction, and one cell in y-direction. In most steps, the particle appears as a combination of small pieces in 2-4 cells. Although these small pieces are reconstructed separately by PLIC algorithm in each cell, they are always connected without splitting. It is emphasized that the translation is realized simply by a typical PLIC algorithm with the surface normal defined in the cell. No additional treatment is necessary.

Using the PLIC/SN method, for 1-D geometry the subgrid particle actually stays on the upstream side of the cell before the normal vector is reversed, which happens exactly after the arrival of the zero vector. The leap motion in the cell and the volume advection through faces are not coupled. It is different for a general 2-D geometry. Although the sign of l is reversed between Fig. 4d and f, the orientation and location of the reconstructed interface are evolved gradually as shown from Fig. 4b to f. The volume flux through the right face is not zero in Fig. 4e, so that a small portion is advected to the right cell as seen in Fig. 4f. The same is seen in y direction from Fig. 4h to j when the sign of m is reversed. As a result of this coupling between the reconstruction and the volume advection, the subgrid particle is located in favor of the downstream sides, which is clearly seen in Figs. 4p and q. As interpreted by Černe et al. (2002), using a typical PLIC-VOF method, the location error will be accumulated, resulting in the particle moving faster.

In order to evaluate the accuracy of the motion of a subgrid particle in two dimensions, three directions are tested. A particle of $d/\Delta x = 0.1$ is initially located at (0.25,0.25), and put in three velocity fields, $(u, v) =$

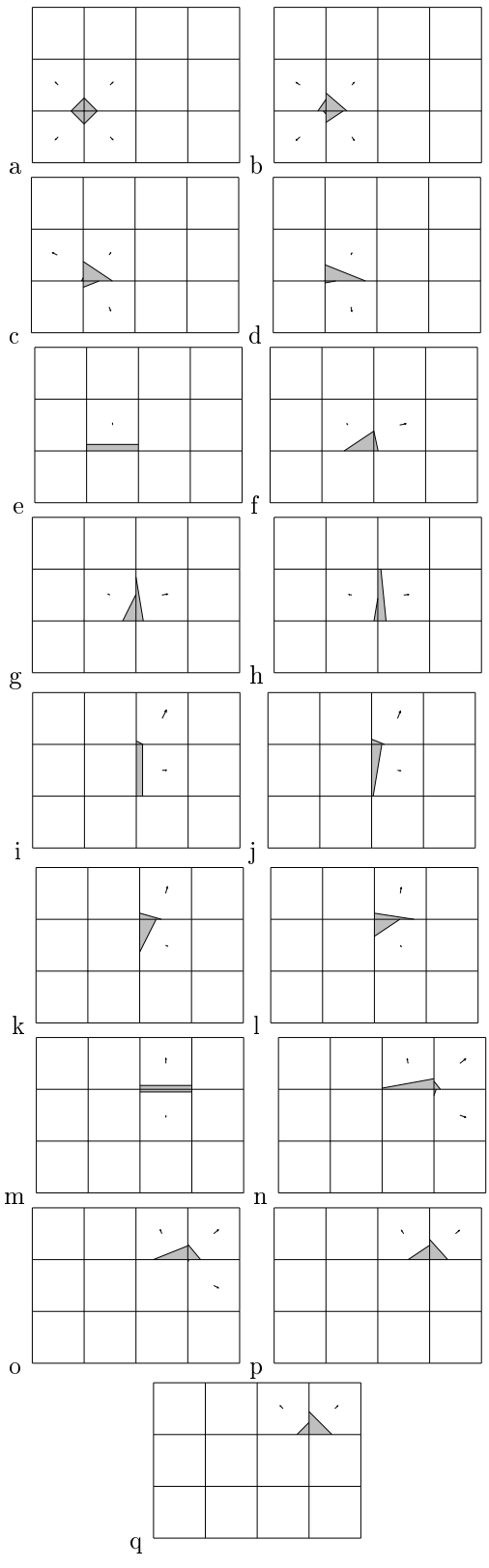

Fig. 4: Translation of a 2-D subgrid particle of $d/\Delta x = 0.4$ in the velocity field $(u = 1, v = 0.5)$, all sequential steps. The particle is supposed to move from one node to the other node in 16 steps, traveling two grid cells in x-direction, and one cell in y-direction.

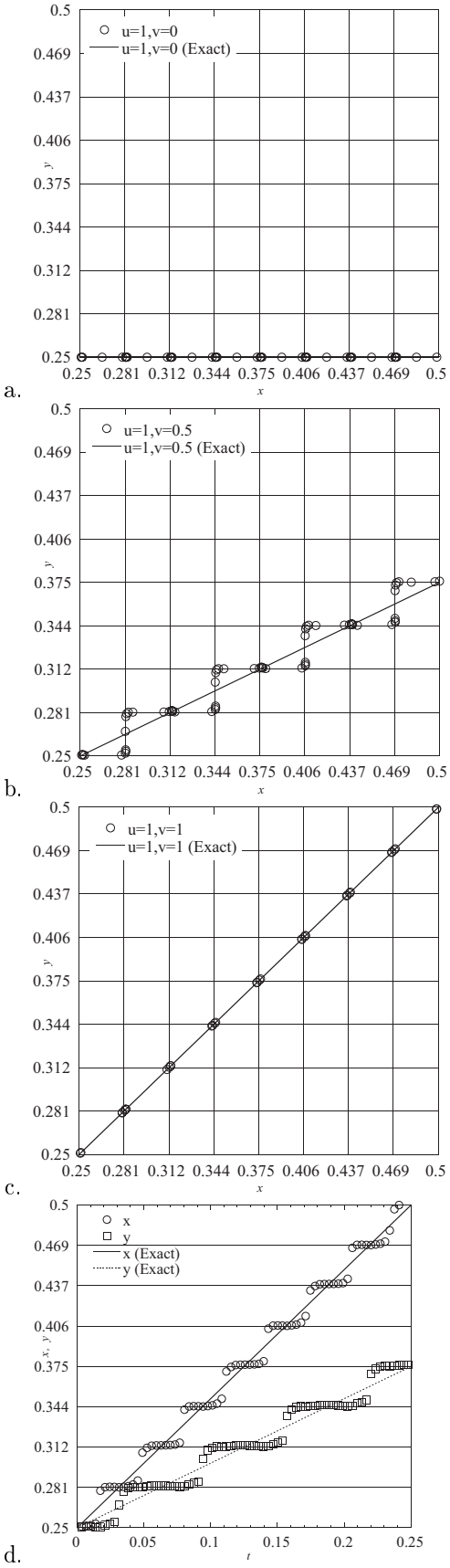

Fig. 5: Trajectories of a subgrid particle ($d/\Delta x = 0.1$) in thee velocity fields: (a) $(u, v) = (1.0, 0.0)$; (b)$(u, v) = (1.0, 0.5)$; (c) $(u, v) = (1.0, 1.0)$; (d) the historic location (x, y) of the particle in velocity field (b).

Tab. 1: Location errors for particles in three velocity fields after translating over 16 cells in x-drection. The errors are normalized by grid size Δx.

$(d/\Delta x)$	$(u, v) = (1.0, 0.0)$	$(1.0, 0.5)$	$(1.0, 1.0)$
0.1	0.030	0.059	0.059
0.4	0.110	0.175	0.184
1.6	0.204	0.162	0.184
6.4	0.013	0.182	0.010

$(1.0, 0.0)$, $(1.0, 0.5)$ and $(1.0, 1.0)$. They represent three typical directions on a square mesh as shown in Figs. 5abc. The centroid location of the particle at every time step is plotted, so these discrete points form the trajectory of the particle motion. For evaluating the location of a particle, the centroid of its portion, $\mathbf{r}_{cj}$, in every cell is first calculated, and then the centroid of the whole particle, $\mathbf{r}_c$, is the area-weighted value,

$$\mathbf{r}_c = \frac{1}{\sum \Omega_j} \sum_j (\Omega_j \mathbf{r}_{cj}),$$

where Ω_j is the particle volume in cell j. The formula is exact for arbitrarily shaped particles.

It is seen that the particle moves along the grid lines nearest to the exact trajectory, and gathers around the grid nodes. Although it follows the exact solution, as shown in Figs. 5ac in space, it actually does not march with the exact one in time. A space-time figure is shown in Fig. 5d for velocity $(u, v) = (1.0, 0.5)$. Since the same velocity is used in x-direction, the historic x-location is very similar for all three velocities. The particle moves in small steps from one node to the other. A periodic location error of $\Delta x/2$ can be seen.

The trajectories for different sized particles are also compared, and plotted in Fig. 6. The diameter is varied from 0.1 to $12.8\Delta x$, covering the whole range from under-resolved to resolved scales. Two types of location error can be identified. One is the 'leap' error due to the interface reconstruction, which dominates in the motion of very small subgrid particles, as shown by the stepwise solid line with circles for $d/\Delta x = 0.1$. Another is the error resulting from the volume advection algorithm, as shown by the dashed line for $d/\Delta x = 12.8$. The advection error is much smaller and with a high frequency. For the particles between, these two errors are coupled.

The location error of particles for all velocities is measured at $t = 0.5$, after the particle has traveled exactly 16 cells in x-direction. The error is defined as the distance between the numerical location and the exact one. Errors normalized by the grid size Δx are recorded in Table 1. At $t = 0.5$, the particle is supposed to exactly locate at a grid node, so the periodic error is the minimum. The data mainly reflects the errors from the advection of volume fluxes. For subgrid particles of $d/\Delta x < 1$, the location error increases with the diameter. For the large particle of $d/\Delta x = 6.4$, the error varies significantly with the flow direction, because of the naive unsplit advection algorithm used. Nevertheless, all errors are within a small fraction of one grid spacing. It is clear that the location error is not accumulated using the PLIC/SN method, or all sized particles are advected at a right speed.

By analyzing the results above, the location error of a subgrid particle can be estimated as,

$$\epsilon_{\max} = \pm\Delta x/2 + d + O(\Delta x^2), \qquad (28)$$

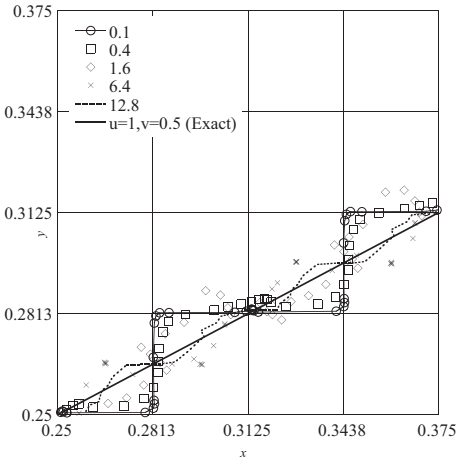

Fig. 6: Trajectories of particles with diameter $d/\Delta x$ varying from 0.1 to 12.8 moving at velocity of $(u, v) = (1.0, 0.5)$. The first two cycles are shown. In each cycle, the particle is advected by two grids in x direction, and one grid in y direction.

where three errors are originated from the interface reconstruction, the advection and the surface normal equations respectively. The first is the leap error. The location of a particle within the cell is solely determined by the PLIC together with the normal vector. The subgrid particle is reconstructed either on the left or the right side depending upon the location of zero vector. If the zero vector represents the exact location of the particle, the location error originated from this reconstruction procedure is no more than $\Delta x/2$, which is the first term. The error originated from the advection algorithm will depend on the algorithm used and the location of interface, so d is taken as the simple estimate of the maximum error, if it is smaller than the grid spacing. The numerical error in integrating the surface normal equations is assumed to be second order accurate. The deviation of the exact center of the particle from the zero vector is $O(d^2)$ as discussed in section 3., which is smaller than $O(\Delta x^2)$ for a subgrid particle. The last term $O(\Delta x^2)$ represents these two errors. All numerical results above show that the location errors in two dimensions are within the bound (28).

It is noted that a more precise location of the subgrid particle can be reconstructed in the cell, by taking into account the location of the zero vector that can be obtained from the distribution of the normal vector. However, this necessitates non-trivial modifications in PLIC and advection algorithms.

6. Concluding remarks

The PLIC/SN method introduced here in the PLIC-VOF methods is unique in that the surface normal vector is directly solved by integrating the equations for it. Given point-valued surface normals, the method provides better, at least comparable, accuracy for particles at low to medium resolutions, although the less accurate advection algorithm (DDR) is adopted in this work. Grid convergence has been obtained for particles with a diameter of few grid cells, which are generally regarded as under-resolved. It is of interest to see that a subgrid particle can be advected by using the method. The zero surface normal moves at the speed of a subgrid particle. The particle is numerically reconstructed following the zero surface normal in the PLIC/SN method, so that it can be advected at the correct speed. Therefore, the method is standalone to resolve and track *any* sized particles.

Two sets of equations for surface normal vector have been investigated and compared. One is conservative, and the other is not. The first-order and the second-order upwind schemes are used to integrate both equations. For the second-order scheme, the influence of the limiter is also investigated. It is found that a high-order scheme based on the non-conservative equations is most robust, and generally provides the best accuracy. The limiter reduces the accuracy of the second order schemes in resolving smooth interfaces, and for solving the conservative equations it can be catastrophic even for smooth interfaces.

The implementation of the method in existing PLIC-VOF codes is quite straightforward. One needs only to replace the surface normal calculation in the PLIC by using the surface normal vector solved by the equations. A better accuracy at medium and high resolutions is expected if coupled with an advection algorithm with corner flux treatment.

As far as computational cost is concerned, it is seemly that the method solving two equations in the whole domain is more expensive than the traditional method with only local operations. We shall consider two cases. If the domain is covered by many particles or interfaces, there will be no much difference between a global and a local operation. In this case, the interfaces are generally resolved by a coarse grid. The accuracy for particles at low grid resolution is a more important issue. On the other hand, if the domain is covered by very few interfaces, the narrow-band idea developed in level-set methods would be useful. However, for both cases, a general way to enhance the efficiency is to couple the PLIC/SN method with a solution-adaptive grid technique. The overhead required to integrate equations on the portion of coarse grids is just a small fraction of total computation.

Reference

(1) W.J. Rider, D.B. Kothe, Reconstructing volume tracking, J. Comput. Phys. **141** (1998) 112-152.

(2) R. Scardovelli, S. Zaleski, Analytical relations connecting linear interfaces and volume fractions in rectangular grids, J. Comput. Phys. **31** (2000) 228-237.

(3) D.L. Youngs, An interface tracking method for a 3D Eulerian hydrodynamics code, Technical Report 44/92/35, AWRE, 1984

(4) E.G. Puckett, A volume of fluid interface tracking algorithm with applications to computing shock wave rarefraction, Proceedings of the 4th International Symposium on Computational Fluid Dynamics, 1991.

(5) G. Černe, S. Petelin, I. Tiselj Numerical errors of the volume-of-fluid interface tracking algorithm, Int. J. Numer. Meth. Fluids, **38**, (2002) 329-350.

(6) J. López, J. Hernández, P. Gómez and F. Faura, An improved PLIC-VOF method for tracking thin fluid structures in incompressible two-phase flows, J. Comput. Phys. **208** (2005) 51-74.

(7) M. Raessi, J. Mostaghimi, M. Bussmann, Advecting normal vectors: a new method for calculating interface normals and curvatures when modeling two-phase flows, J. Comput. Phys. **226** (2007) 774-794.

(8) 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) 151-172.

(9) M. Sun, K. Takayama, Conservative smoothing on an adaptive quadrilateral grid, *J. Comput. Phys.* **150** (1999) 143-180.