INTERFACE RECONSTRUCTION WITH SPLIT LAGRANGIAN ADVECTION FOR TWO-DIMENSIONAL INTERFACIAL FLOWS

C. S. Wu D. L. Young *

Department of Civil Engineering and Hydrotech Research Institute
National Taiwan University
Taipei, Taiwan 10617, R.O.C.

ABSTRACT

This paper contributes to propose a 2D practical interface tracking algorithm of volume-of-fluid (VOF) method through the improved interface reconstruction, namely using both mixed Youngs centered column method (MYCCM) and the Lagrangian split advection scheme. A computationally efficiency and second-order accuracy for interface reconstruction method is presented and approximated by heuristic algorithms based on the piecewise linear interface calculation (PLIC) concept. The method can be accurately estimated by a regular structured mesh without any geometrical modifications. Besides, a linear mapping technique is implemented to improve the efficiency of numerical simulations with regard to the approximation for capturing the interface. The computational tests include widely used benchmark cases, such as the solid-body translations and rotations and the swirled single vortex of fluid body. Its performances of the improved algorithm are compared against other classical VOF advection methods. Good results are obtained by using present algorithm.

Keywords: Volume-of-fluid, Interface tracking, Advection, Mixed Youngs centered column method, Piecewise linear interface reconstruction.

1. INTRODUCTION

One of the important subjects that have been considered in fluid mechanics is to study the interfacial flows. Interfacial flows are ubiquitous in many nature and industrial processes, such as splashes, droplets, castings, polymer blending, water waves, ink-jet printers, and so on. Interface tracking for modeling interface kinematics on computational meshes, is required for the accurate simulations of interfacial physics, especially for micro-mechanic problems. Many algorithms have been devised to track and advect the interface with a formulation varying from algebraic to geometric or heuristic system in mathematics. Volume tracking methods are the most popular, because of their established conservation, robustness, and versatility properties, with the achievable property of local volume conservation being particularly valued.

Among the popular tracking approaches which have been proposed and have undergone continual improvement, the major formulations have been developed in multi-phase simulations; they are the surface-marker method (front-tracking method) [1,2], the level-set method [3,4] and the volume-of-fluid (VOF) method [5-8].

In this study, the way to mark the fluid is to use an indicator function, known as the volume fraction or scalar function, which is generally referred as the VOF method. The function flagged the volume fraction in a local cell volume occupied by the fluid. The transport of the indicator function through the domain is governed by a hyperbolic equation. A variety of techniques have been developed to maintain a well defined interface within the frame of the VOF method. The VOF method employed in this numerical study has a unique advantage with its potential to conserve volume. And it is hard to find a scheme that satisfies the properties exactly in numerical simulation due to the limitation to its potential coming from the numerical methods in use and the complicated procedures, especially for the VOF method. In one class of the above mentioned schemes, the accurate location of the front is discarded and the interface is reconstructed in an approximate manner based on the local fluid volume. Recently, all these methods can be combined in a mixed algorithm [9-11] which gets rid of their original drawbacks and disadvantages.

The simulated procedures of the VOF, which is more complicated than other schemes except volume conservation, are categorized into two steps: (1) interface reconstruction and (2) fluid advection. The VOF method solves a non-diffusive solution of the advection equation by a geometrically based calculation, with the void fraction flows at the cell faces based on the reconstructed interface. The reconstruction is not a unique process, gives a spatial distribution of volume fraction,

* Corresponding author (dlyoung@ntu.edu.tw)

DOI : 10.1017/jmech.2012.144
Copyright © 2013 The Society of Theoretical and Applied Mechanics, R.O.C.
https://doi.org/10.1017/jmech.2012.144 Published online by Cambridge University Press

and the interfacial geometry depends on the assumed reconstruction algorithm. Two early approaches are the donor-acceptor (DA) algorithm [6] and the simple line interface calculation (SLIC) algorithm [12], in which the interface is represented by a piecewise-constant line in each interfacial cell, either vertically or horizontally. A significant improvement of interface representation is the piecewise linear interface calculation (PLIC) scheme [13]. This scheme is approximated by a straight line of arbitrary orientation in each cell. The interface is described by a segment perpendicular to the gradient of a scalar function. The position of this inclined segment in each cell is uniquely determined from volume conservation. The orientation is to find the distribution of one of the fluid in the neighboring cells, but the method does not reconstruct all the linear interfaces exactly.

The detailed reviews of interface reconstruction methods corresponding to the DA, SLIC and PLIC can be found in Rider and Kothe [14,15]. Then, the least-squares VOF interface reconstruction algorithm (LVIRA) [16] is developed to improve the drawbacks of the PLIC scheme. Under the constraint of the reconstructed volume we have the same volume in a cell. This method uses the linear function in a way to minimize the reconstruction errors by a 3 × 3 stencil centered at this cell. After that, to reduce the computational time from LVIRA, an efficient LVIRA (ELVIRA) scheme is improved [17]. The ELVIRA scheme calculated interface normal by heuristic criteria via the combination of backward, central and forward finite-differentiations of volume fraction. The residual is then calculated for nine candidates and the one that has the smallest residual is chosen as the optimal solution. From the above mentioned schemes, further improvement is still needed, especially as encountering the interface which contains large curvature, singular points and filaments structures. Attempting to overcome these shortcomings, the polygon area mapping (PAM) is proposed [18]. This method has very little spatial numerical diffusion and tracks the interface singularities naturally without any modifications within numerical analysis. However the CPU-time cost of the PAM method increases as the velocity fields get more complicated. To maintain both acceptable accuracy and high efficiency of the interface reconstruction, an improved PLIC scheme, which combines both the Youngs method [14] and the centered column method [19,20] or called the MYCCM, is employed in our algorithm.

As far as the second step of fluid advection is concerned in this study, the Lagrangian split method which updates the volume fraction by advecting the interface along one spatial direction in a time increment is employed. In this study, the proposed interface tracking method combined with the Lagrangian split advection method and the MYCCM is implemented to perform the interfacial flows in the fluids.

2. METHODOLOGY OF 2D INTERFACE TRACKING

The classical VOF method, at each time step, offers the volume fraction of one of the two immiscible fluids in each computational cell and estimates the normal vector to the interface. The interface of 2D space in the fluid is approximated and defined in each cell by a straight segment. The interface constructed within the cell having the same normal and dividing the cell into two parts is then propagated by the flow. Thereby the resulting volume, mass, momentum fluxes of each fluid into neighboring cells are determined. The updated values of the volume fraction in each time step show the variation of fluid body throughout the computational domain. The following subsections describe the procedures for estimating the construction of planar surface in each cell, the normal vectors, and the interface propagation by the flow.

2.1 Two-Dimensional Interface Reconstruction Algorithm

2.1.1 Expressions for the Cut Area

In two dimensions, the intersection within the cell described by a line with a number of sides, total amount to four types is schematically represented in Fig. 1. The interface between materials is approximated by a linear equation in each computational cell with normal **m** which is defined as

$$\mathbf{m} \cdot \mathbf{x} = \alpha \ , \tag{1}$$

in which the vector, $\mathbf{m} = (m_1, m_2)$, normal to the interface points outside the region towards the different phase where the volume fraction is zero. $\mathbf{x} = (x_1, x_2)$ and α is a parameter related to the smallest distance between the line and the origin. The problem of finding the parameter α is complicate. Before doing that, two main issues require to consider; that is, how to treat the "forward" and "inverse" relations in the process of interface reconstruction. We distinguish the "forward" problem, that is, to find the volume fraction occupied by one species by given **m** and α. As stated, the problem is essentially geometric in nature. Several implementations use extensively embedded *if-elseif-endif* constructions in the programming. In some cases, large number of levels in these constructions and number of cases make the code difficult to compile optimally and to maintain. As for the "inverse" problem, it consists of finding α by given the fraction of volume contained on each side and the normal direction. In this study, we offer an efficient way to overcome the problem by the linear mapping technique. The relation between volume fraction and α is continuous and one-to-one. Both the forward and inverse problems

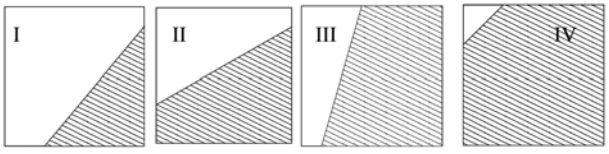


Fig. 1 The four types of interface cut by a 2D line segment

https://doi.org/10.1017/jmech.2012.144 Published online by Cambridge University Press

can occur several times, according to the chosen numerical schemes, during the reconstruction and propagation of the interface. Once the normal direction and cut volume are known, then the value α is computed by enforcing volume conservation. Geometrically the constraint is applied by moving the interface plane along the normal direction, changing the free parameter until the area under the line is equal to the computed area.

In the standard forward problem we assume that the normal directions are all positive and we need to determine the cut area of the rectangular cell which is below the interface as shown in Fig. 2. The cell with sides δx_1 and δx_2, and the straight line with normal vector **m** are given to find the corresponding area of the region *ABFGD* which lies within the rectangular cell. An expression for this area depends on the positive normal direction and the area of region contained below the straight line within the rectangular *ABCD* which is given by

$$A = \frac{\alpha^2}{2m_1 m_2}\left[1 - H(\alpha - m_1 \delta x_1)\left(\frac{\alpha - m_1 \delta x_1}{\alpha}\right)^2 - H(\alpha - m_2 \delta x_2)\left(\frac{\alpha - m_2 \delta x_2}{\alpha}\right)^2\right], \quad (2)$$

where $H(\bullet)$ is the Heaviside step function. The points, *E* and *H*, at which the line intersects with the axes are respectively at α/m_1 and α/m_2. In case points *E* and *H* lie within the original rectangle, the desired area is the first term of Eq. (2). As the point *E* lies outside the considered region over the point *B*, that is $\alpha > m_1 \delta x_1$, the triangle area *BEF* must be subtracted from the desired region to obtain the proper area. Since triangle *BEF* is geometrically similar to *AEH*, the ratio of these two areas is equal to the square of the ratio of the sides *BE* to *AE*, and given by

$$\frac{A_{BEF}}{A_{AEH}} = \left(\frac{\alpha - m_1 \delta x_1}{\alpha}\right)^2 . \quad (3)$$

Similarly, as the point *H* lies over the point *D*, that is $\alpha > m_2 \delta x_2$, the triangular area *DHG* should be subtracted from the considered region as the third term of Eq. (2).

As for the inverse problem, the parameter α is required to be determined corresponding to a given cut area and known normal directions in a computational cell. There are a number of ways to achieve this requirement. One can simply use a standard root-finding approach (for example: bisection method or Newton's method) to find the parameter α at which the cut area has reached the desired value. To obtain more efficiency and less difficulty of the numerical studies, the linear-mapping process is added and employed in this numerical model instead of the iterative method as used by Rider and Kothe [14]. In two dimensions, the type of cases can be compiled into four situations as shown in Fig. 3. The four cases should be mapped to

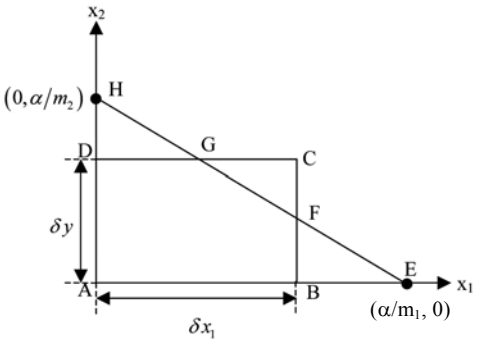


Fig. 2 The 2D sketch of cut area under a straight line

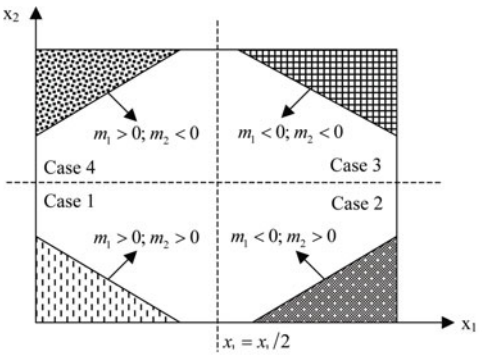


Fig. 3 The normal estimation by linear mapping procedure with four situations

the Case 1 ($m_1 > 0$, $m_2 > 0$) and the description of interface should satisfy Eq. (1). We take the Case 2 for illustration ($m_1 < 0$, $m_2 > 0$), the linear equation of the interface can be reflected to the Case 1 along the line $x_1 = \delta x_1/2$. By the linear transformation, we can find the relation, $x_1' = \delta x_1 - x_1$, and then substitute it into the original linear equation in Eq. (1). The linear equation of interface in Case 2 can be described, and the parameter is transformed into $\alpha' = \alpha - m_1 \delta x_1$. Similarly, the linear equations of Case 3 ($m_1 < 0$, $m_2 < 0$) and Case 4 ($m_1 > 0$, $m_2 < 0$) can be determined by the same concept. The parameter α can be transformed as $\alpha' = \alpha - m_2 \delta x_2$ and $\alpha' = \alpha - m_1 \delta x_1 - m_2 \delta x_2$, respectively. The analytical expressions for both the forward and inverse relations can be derived with some straightforward algebra as follows:

For the 2D forward problem we obtain

$$A = \frac{\alpha^2}{2m(1-m)} \quad \text{for} \quad 0 \le \alpha < m , \quad (4)$$

$$A = \frac{2\alpha - m}{2(1-m)} \quad \text{for} \quad m \le \alpha \le \frac{1}{2} , \quad (5)$$

where $m = \min(m_1, m_2)$.

For the 2D inverse problem we obtain

$$\alpha = \sqrt{2m(1-m)A} \quad \text{for} \quad 0 \le A < A' , \quad (6)$$

$$\alpha = A(1-m) + \frac{m}{2} \quad \text{for} \quad A' \le A \le \frac{1}{2} , \quad (7)$$

https://doi.org/10.1017/jmech.2012.144 Published online by Cambridge University Press

where $A' = m/2(1-m)$. Notice that the limit $m = 0$ is correctly described by these expressions.

2.1.2 Normal Estimation

We consider a 2D computational domain with square cells of side h. The fractional function F_{ij} represents the discrete version of the area fraction as follows:

$$F_{ij}(t) = \frac{1}{A_{ij}} \int_{A_{ij}} \zeta(\mathbf{x},\ t)\, d\mathbf{x}\ , \tag{8}$$

where A_{ij} is the area of the computational grid cell and ζ is the 2D scalar function. In this approach, based on the classical PLIC-VOF concept, the interface is represented in each cut cell by a portion of a straight segment, perpendicular to the location of volume fraction. The reconstruction of a normal direction to the interface is estimated by finite-difference approximations and may satisfy some other minimizing criteria as follows:

$$\mathbf{m} = -\nabla F\ . \tag{9}$$

The cell corner value of the normal vector $\mathbf{m}$ is computed and illustrated by an example to define the normal vector at point $(i + 1/2, j + 1/2)$ as follows:

$$m^x_{i+1/2,\ j+1/2} = \frac{1}{2h}(F_{i+1,\ j} - F_{i,\ j} + F_{i+1,\ j+1} - F_{i,\ j+1}) \tag{10}$$

$$m^y_{i+1/2,\ j+1/2} = \frac{1}{2h}(F_{i,\ j+1} - F_{i,\ j} + F_{i+1,\ j+1} - F_{i+1,\ j})\ . \tag{11}$$

In two dimensions, the method is first-order accurate with a nine-point finite difference approximation of the gradient as referenced and proved by Gueyffier *et al* [5]. The normal vector requires the four cell-corner values to be computed and averaged as the cell-centered value in each cell as follows:

$$\mathbf{m}_{ij} = \frac{1}{4}(\mathbf{m}_{i+1/2,j-1/2} + \mathbf{m}_{i-1/2,j-1/2} + \mathbf{m}_{i+1/2,j+1/2} + \mathbf{m}_{i-1/2,j+1/2})\ . \tag{12}$$

In this study, we choose a different way to define the normal direction based on some heuristic criteria and originally based on the PLIC scheme. This scheme depending on the centered columns concept is partially similar to the ELVIRA algorithm [17]. The local height function or local width function of the interface, $y = f(x)$ or $x = g(y)$, is defined by the volume fraction values in a 3 × 3 square blocks as schematically shown in Fig. 4. In Fig. 4(a), the height y_{i-1} at the abscissa x_{i-1}, placed in the center of column, is given by the following expression:

$$y_{i-1} = h\sum_{k=-1}^{1} F_{i-1,\ j+k}\ . \tag{13}$$

The linear function of interface can be described as

$$\mathrm{sgn}(m_2)x_2 = m_1\, x_1 + \alpha^*\ , \tag{14}$$

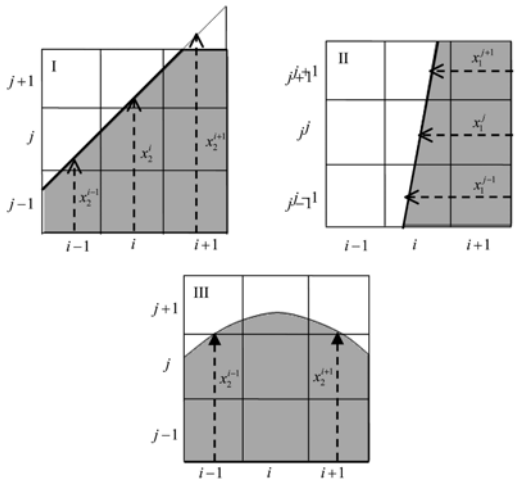


Fig. 4 The volume fractions are added column-wise, row-wise and curved expression of a linear interface

where $m_2 = -\partial F/\partial x_2$. The sign of m_2 is required to get the height function due to lose track of phase point inside or outside as we integrate the volume fraction. The sign and angular coefficient m_1 can be approximately defined by backward, forward or central finite-differences respectively. They are

$$m_1^f = \frac{x_2^{i+1} - x_2^i}{h}\ , \tag{15}$$

$$m_1^b = \frac{x_2^i - x_2^{i-1}}{h}\ , \tag{16}$$

$$m_1^c = \frac{x_2^{i+1} - x_2^{i-1}}{2h} = \frac{1}{2h}\sum_{k=-1}^{1}(F_{i+1,j+k} - F_{i-1,j+k})\ . \tag{17}$$

In Fig. 4(a), the interface cuts the two opposite sides of the column, since the two local heights x_2^i and x_2^{i-1} are on the interface, the coefficient $m_1 = m_1^b$. In case of the column $(i + 1)$, the height is not on the interface and outside the considered area. The small area outside the block of cells is missing and the forward and central finite-difference underestimate the coefficient m_1. Therefore, two estimates, say m_{x1} and m_{x2}, are available for the same coefficient m_1, we select one of them to be the slope of linear interface with the following criterion

$$|m^*| = \max\left(|m_{x1}|, |m_{x2}|\right)\ . \tag{18}$$

Similarly, the width function in Fig. 4(b) can also be defined by the linear equation

$$\mathrm{sgn}(m_1)x_1 = m_2\, x_2 + \alpha^{**}\ . \tag{19}$$

For the almost vertical case in Fig. 4(b), the three finite-difference scheme can calculate the correct parameter m_2, as the line cuts two opposite sides of each row. As the line of interface becomes vertical, a rather

https://doi.org/10.1017/jmech.2012.144 Published online by Cambridge University Press

crude approximation of the value m_2 should go to infinity. Therefore, another heuristic criterion between two approximations, m_{x1} and m_{y1}, should be taken as follows:

$$| m^* | = \min\left(| m_{x1} |, | m_{y1} | \right) . \tag{20}$$

The above criteria optimize the reconstruction of a linear interface, but we may not get the optimal value for the curved interface as the interface shown in Fig. 4(c). For this case, we get the best approximation since $m_2^c = 0$, but this would require the minimal value in Eq. (18). However, the two criteria can optimize the better guess of normal estimation. From the above mentioned, we can obtain the accurate approximation of the normal direction which depends on the known volume fractions by adding column-wise or row-wise volume fractions instead of using cell-centered values completely estimated by the PLIC.

2.2 Lagrangian Split Advection Method

The Lagrangian method instead of the Eulerian method can compute directly the motion of the interface segments as schematically in Fig. 5. From Eq. (1), the interface of linear equation at the initial time t^n can be rewritten with superscript n attached to all the variables as follows:

$$m_1^n x_1^n + m_2^n x_2^n = \alpha^n . \tag{21}$$

Lagrangian advection of this interface by the flow as the time increases to t^{n+1} will move it to a new position within the time increment Δt which must be calculated. In practice, the time step is performed separately in each spatial direction. In two dimensions, to eliminate the asymmetric behavior of the advected fluid body, the sequence of the advection follows the rules: $(x_1 \rightarrow x_2)$ directions $\rightarrow (x_2 \rightarrow x_1)$ directions to maintain the symmetry of fluid body in a two time-steps construction. To make the description of interface simpler, we take the x_1 direction advection for example; let us suppose the considered 2D cell with the velocities on the left and right faces by $u_{i-1,j}$ and $u_{i,j}$ respectively. These are taken to be constant over the entire face to which they are assigned. If the fluid body moves in x_1 direction, the component of velocity is described by a linear interpolation form

$$u(x_1) = u_{i-1,\,j}\left(1 - \frac{x_1}{\delta x_1}\right) + u_{i,\,j}\frac{x_1}{\delta x_1} . \tag{22}$$

For each point initially at x_1^n as shown in Fig. 5, the velocity is calculated and assumed to remain the constant during the advection step. Then each point initially on the fictitious interface as described in Eq. (22) will change to a new provisory position

$$\begin{aligned} x_1^* &= x_1^n + u(x_1)\,\delta t \\ &= \left[1 + \left(\frac{u_{i,\,j} - u_{i-1,\,j}}{\delta x_1}\right)\delta t\right] x_1^n + u_{i-1,\,j}\,\delta t , \end{aligned} \tag{23}$$

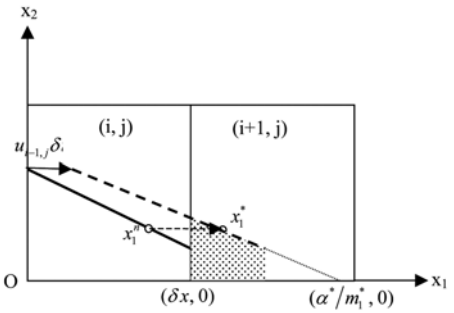


Fig. 5 The schematic description of split Lagrangian propagation in x_1 direction. The bold line of segment is the original interface; the dashed one is the position after advection. The volume gained by the downwind cell is the shaded volume

where the superscript $*$ denotes a fractional step rather than t^{n+1}. We can have an intermediate expression of x_1^n as a function of x_1^* as follows:

$$x_1^n = \frac{x_1^* - u_{i-1,\,j}\delta t}{1 + \left(\dfrac{u_{i,\,j} - u_{i-1,\,j}}{\delta x_1}\right)\delta t} . \tag{24}$$

The x_2 coordinate remains constant during this advection. The similar step in x_2 direction is followed before the advection to the next time step. And we substitute it into Eq. (21) for the interface after advection, we have

$$m_1^n \frac{x_1^* - u_{i-1,\,j}\,\delta t}{1 + \left(\dfrac{u_{i,\,j} - u_{i-1,\,j}}{\delta x_1}\right)\delta t} + m_2^n\, x_2^n = \alpha^n . \tag{25}$$

The linear equation can be written in a more standard form

$$m_1^* x_1^* + m_2^* x_2^* = \alpha^* , \tag{26}$$

where

$$m_1^* = \frac{m_1^n}{1 + \left(\dfrac{u_{i,\,j} - u_{i-1,\,j}}{\delta x_1}\right)\delta t} ; \quad m_2^* = m_2^n \tag{27}$$

and

$$\alpha^* = \alpha^n + \frac{m_1^n u_{i-1,\,j}\delta t}{1 + \left(\dfrac{u_{i,\,j} - u_{i-1,\,j}}{\delta x_1}\right)\delta t} . \tag{28}$$

Similarly, the same process of advection can be analyzed in the x_2 direction. After the advection for each cell, the area that the interface protrudes to the neighboring cells should be calculated. Then, we make an additional coordinate transformation of the interface in Eq. (26), at which the origin of the cell has changed to the point (δx_1, 0). The description of the

https://doi.org/10.1017/jmech.2012.144 Published online by Cambridge University Press

linear interface which was moved to the right neighboring cell has been rewritten as

$$x_1^* = \delta x_1 + x_1' \\ \alpha' = \alpha^* - m_1^* \delta x_1 \ . \tag{29}$$

Then, the forward problem that we mentioned before is employed to calculate the volume of fluid body moving out or remaining in each cell.

3. NUMERICAL TESTS FOR VOLUME TRACKING ALGORITHM

In this section the performance of the improved PLIC-VOF scheme is shown over standard numerical tests. First, we consider simple flows such as uniform translations and rotations where a smooth fluid body should be advected without distortion and the mass should be conserved exactly. We then analyze the performance of the improved PLIC-VOF algorithms with the flow field enforced by the non-uniform vorticity including the single vortex flow. The mass error of all the testing cases is determined by the following relative error analysis at two instants (t_0, t_1),

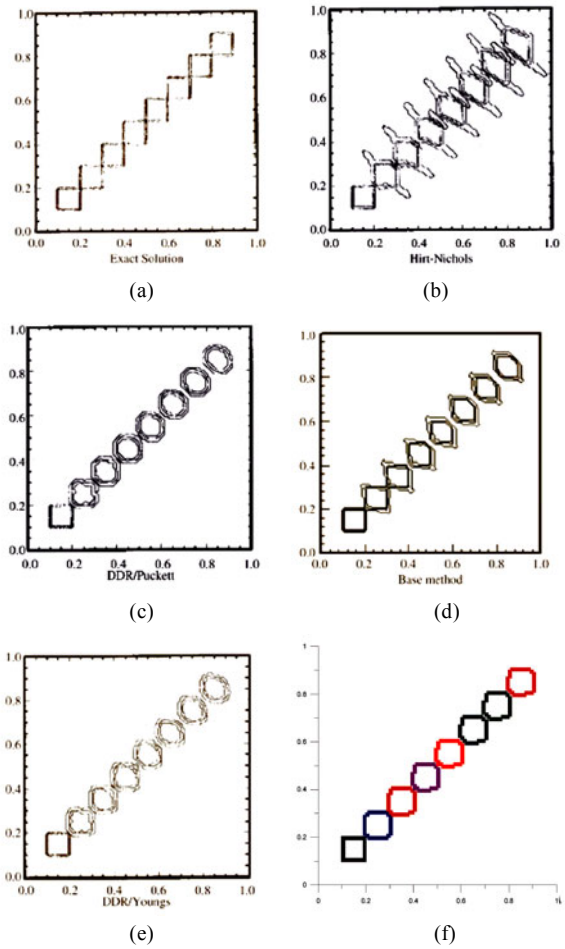
$$E_m = \frac{\sum_{ij} \left\| F_{ij}(t_0) - F_{ij}(t_1) \right\|}{\sum_{ij} F_{ij}(t_0)} \tag{30}$$

The total volume occupied by the referenced phase at the initial time t_0 and that of the verified fluid body at time t_1.

3.1 Two-Dimensional Translation Test of a Box

It is a simple test with translating a geometric shape around the computational domain. The geometric shape should remain intact and be conserved for the total volume of fluid within the whole region. The dimension of computational domain is $1 \times 1\text{m}^2$ which is composed of 10,000 equal mesh size. A square block of fluid with the size $0.1 \times 0.1\text{m}^2$ moves toward the top right-hand corner with a uniform and constant velocity, both equal to 1ms^{-1}. The computational time increment is 10^{-3}s leading to a Courant number of 0.1 and the result of fluid position is shown every 0.1s until 0.7s. Under this test, the result is examined to highlight some of problems existing with conventional H-N algorithm [6], DDR algorithm [21] and base method [22] (derived by the H-N model without correction for the diagonal neighbor cells).

In Fig. 6, we show the VOF contour calculated by the H-N algorithm, DDR, DDAR algorithms, base method, the proposed improved VOF scheme and the exact solution. In Fig. 6(b), the H-N algorithm shows significant diffusion of fluid in a direction normal to the velocity of fluid. Also, there is wisp generation evident at the new box corner and the shape of the box at the end "flattened" to the extent. Besides, some diffusion of the box shape still occurs by the DDR algorithm



(a) (b) (c) (d) (e) (f)

Fig. 6 The evolution for the advection of the box (Some images are derived from [22])

and base method as shown in Figs. 6(c) ~ 6(e). This is evidenced by the rounding of the box corners and some spreading of the box in the direction to the flow path. In comparison, the investigations do not include any suppressing or spreading features in this improved VOF scheme, and the total distance moved by the centre of the box is predicted more accurate than other referenced schemes. Table 1 shows the mass error of each translation test. The DDAR and the improved VOF scheme show better performance than others. In addition to the efficiency, Table 2 shows the total computational unit (CPU) time required to perform the translation test. All data are normalized against the time required by the H-N algorithm. The total CPU time corresponding to the improved VOF scheme consumes only 4.125s.

3.2 Two-Dimensional Translation Tests for Hollow Geometry

Translation tests show the most basic feature of volume advection for interface tracking. In this section, we take three different hollow geometries into consideration with a unidirectional velocity field. The form of the tests here is taken from Rudman [23], they are (1) a hollow square aligned with the coordinate axes, (2) a titled hollow square orientated at 26.57° to the axis and (3) a hollow circle.

https://doi.org/10.1017/jmech.2012.144 Published online by Cambridge University Press

Table 1 The error functions calculated through each translation test of box

Method	H-N	DDR/ Youngs	DDR/ Puckett	DDAR	Current data
Error Function	0.539	0.190	0.216	0.126	0.132

Table 2 The non-dimensionalized total CPU time consumed through the translation of box

Method	H-N	DDR/ Youngs	DDR/ Puckett	DDAR	Current data
CPU-time	1.0	1.0	1.3	–	0.965

The three scalar fields are separately advected with two constant velocity fields, $(u, v) = (0, 1)$ and $(u, v) = (2, 1)$, resulting six different combinations together. The considered computational domain with the dimensions $4 \times 4\text{m}^2$ is composed of 200×200 uniform mesh size. The exterior extent of each of the shapes is initially 40 mesh size ($D = 0.8$m) and the distance between the outer and inner interface is 10 mesh size ($d = 0.4$m). A Courant number of 0.25 is used and advection proceeds for approximately 500 time increments.

Figures 7(a) ~ 7(e) and 7(f) ~ 7(j) show the corresponding current data of initial and final positions compared with the other volume tracking methods with the two known unidirectional velocity fields. Some referenced results can be found from Rudman [23] which are shown here. The solution errors of the interface tracking are quantified and compiled in Table 3. It is noticed that the present algorithm gives the most accurate solutions in 6 out of 8 scenarios. In addition the orders of errors lie between 10^{-3} and 10^{-8}. The mass conservation is satisfied by the improved VOF scheme and there is no need to make modifications at the interface. However, if possible, some rounded corners or cusps features and lost accuracy should be restored at the corner of the square. The reason is that the numerical diffusion results from utilizing the data from neighboring cells and limiting to use the piecewise linear function in reconstructing the material interface [18]. The optimal interface normal may not be selected in the case of a curved or a sharp interface. In addition, the total CPU time in this test consumes nearly 13.9*s* for the two unidirectional velocity fields.

3.3 Zalesak Slotted Disk Rotation

The Zalesak test describes that a slotted circle is rotated through a complete revolution within the computational domain under a pure rotation velocity. The stream function in the velocity field is

$$\psi(x_1, x_2) = \frac{-\omega}{2}\left[(x_1 - x_0)^2 + (x_2 - y_0)^2\right], \qquad (31)$$

where x_0, y_0 are the centre of the rotation and ω is a constant angular velocity. The physical domain, with the dimensions 4×4, is composed of 200×200 mesh cells. The diameter R of the slotted circle centered at

Table 3 The errors of unidirectional velocity fields

Velocity	VOF methods	Square	Square (26.57°)	Circle
(1,0)	SLIC	8.42e-8	1.46e-2	1.30e-2
	Hirt-Nichols	1.03e-8	6.91e-2	4.55e-2
	FCT-VOF	3.89e-8	2.32e-2	1.28e-2
	Youngs	1.08e-3	5.35e-3	3.08e-3
	Stream/Puckett	1.61e-3	4.57e-3	1.42e-3
	DDR/ELVRIA	1.65e-3	4.80e-3	2.98e-3
	Current data	1.79e-8	7.60e-4	1.83e-6
(2,1)	SLIC	1.32e-1	1.08e-1	9.18e-2
	Hirt-Nichols	6.86e-3	1.60e-1	1.90e-1
	FCT-VOF	1.63e-8	8.15e-2	3.99e-2
	Youngs	2.58e-2	3.16e-2	2.98e-2
	Stream/Puckett	3.33e-2	3.15e-2	6.96e-3
	DDR/ELVRIA	1.50e-1	1.45e-1	1.54e-1
	Current data	4.40e-6	7.96e-3	3.11e-6

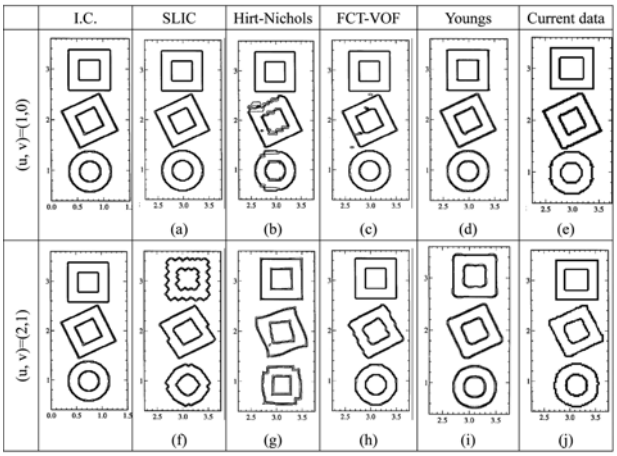


Fig. 7 Advection with unidirectional velocity fields for hollow shape (Some images are derived from [23])

(2.0, 2.75) is 50 mesh cells and the slot width s is 6 mesh cells as depicted in Fig. 8. The axis of rotation is the centre of the computational domain $(x_0, y_0) = (2.0, 2.0)$. The time increments correspond to a Courant number of approximate 0.25 based on the maximum coordinate velocity. A full rotation with constant angular velocity, $\omega = 0.5$, is performed of 2π rotation. One revolution of the disk goes through exactly 2524 time steps, as the slotted circle will return to its initial position.

As shown in Fig. 9, reproduced from Rudman [23], the figure shows the final position of the rotating disk after an evolution calculated using several different advection algorithms. The errors of fluid volume are measured and compiled in Table 4. The proposed improved VOF scheme gets acceptable results with the smallest errors and sharp interfaces without jetsam among the seven methods, although the corner also rounded and lost at the end of the slot. The complete evolution in a full rotation of the solid disk is shown in Fig. 10. Figure 11 shows the reconstruction after a full rotation and compares with the ELVIRA scheme. Some phase errors generated at the corners of the slot propagate to other interface cells and the discontinuities

https://doi.org/10.1017/jmech.2012.144 Published online by Cambridge University Press

Table 4 The errors of the Zalesak slotted rotation disk test

Methods	Error
SLIC	8.38e-2
Hirt-Nichols	9.62e-2
FCT-VOF	3.29e-2
Youngs	1.09e-2
Stream/Puckett	1.00e-2
Stream/Youngs	1.07e-2
DDR/ELVIRA	5.25e-2
Current data	1.027e-2

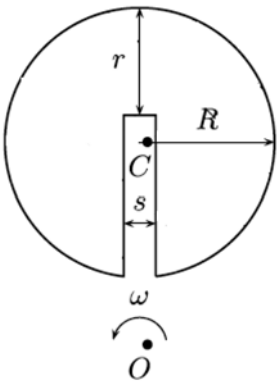


Fig. 8 Sketch of the slotted Zalesak rotation disk

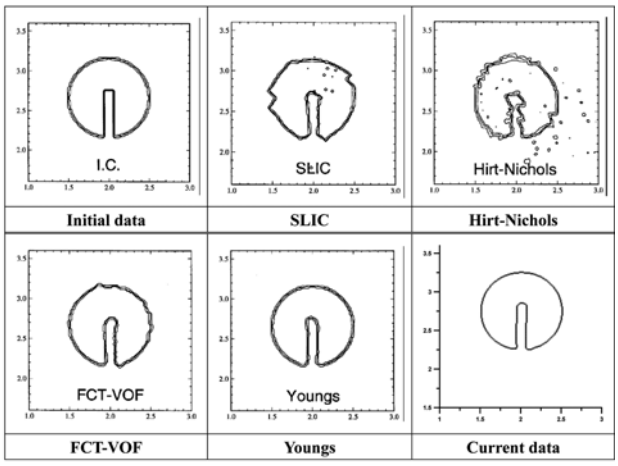


Fig. 9 Final position of Zalesak test for solid body rotation (Some images are derived from [23])

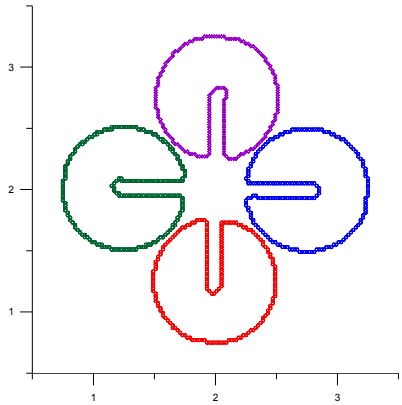
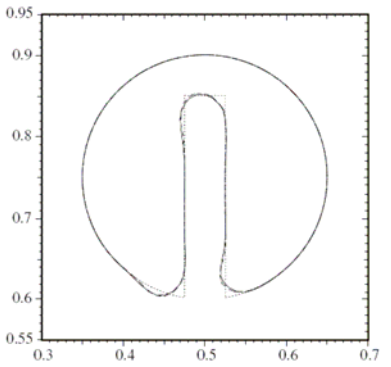
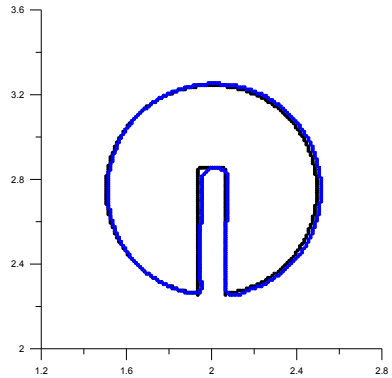


Fig. 10 The solid body advection within a complete rotation



ELVIRA reconstruction



Improved PLIC_VOF

Fig. 11 Zalesak's test for one full solid rotation: the black one is the initial position, the blue line is the final position

are progressively smeared out in time until the radius of curvature can be reconstructed. In addition, the total CPU time costs 26.39*s*. Thus, the Zalesak rotating test presented are largely a measure of the employed interface reconstruction technique to represent high curvatures of fluid interface, rather than those of flux calculation technique. It is significant to note that only minor differences in the errors generated by heuristic piecewise linear algorithm are found.

3.4 Single Vortex Tests

As stated by Rider and Kothe [14], their test cases include only translation of fluid bodies by the velocity field. They suggested that the adequate test cases should include the change of body topology which can be caused by shearing flow. In this test, a non-uniform vorticity field is implemented to analyze the shearing flows with a more precise assessment of reconstruction and advection algorithms.

The computational domain has the dimensions $1 \times 1\text{m}^2$ with uniform grids ($n^2 = 128^2$), and a circle of fluid is centered at the point (0.5, 0.75) with the radius 0.15m. The duration of each test is *Ts*. All boundaries are periodic and the stream function is

$$\psi(x_1, x_2) = \frac{1}{\pi}\sin^2(\pi x_1)\sin^2(\pi x_2)\cos\left(\frac{\pi t}{T}\right). \qquad (32)$$

https://doi.org/10.1017/jmech.2012.144 Published online by Cambridge University Press

The circle of fluid is advected by the solenoidal velocity field, stretching and spiraling about the centre of computational domain. It reaches the maximum deformation at $t = T/2$, and the velocity is reversed by the cosinusoidal temporal factor, then returns to the initial position. Figure 12 shows the reconstructed interface at $t = T/2$ for $T = 0.5$, $T = 2.0$ and $T = 8.0$. The time increment is set to $\Delta t = 1/n$, so the maximum Courant number in this test equals to one. In Fig. 12, it is obvious to observe that the velocity field stretches the interface into filament that spirals toward the centre of the vortex, potentially tearing the material apart if the grids resolution is not fine enough or the interface tracking algorithm is not good enough. Then, a precise indication of the accuracy of the adopted scheme can be deduced by comparing the initial and final position of the fluid circle. In Fig. 12, we compare the results with stream/Youngs method [24] and adaptive moving grids VOF method [10]. All the VOF methods we referred above tend to produce fragmentation of the fluid filament as its thickness becomes to the cell size. The reason is that calculation of the interface normal yields spurious results when there are more than one interface segment inside the same stencil. The information on the interface orientation is lost at the end of advection step since only results retained are the values of volume fractions. Consequently, the choice of the reconstruction step is the source of numerical diffusion and is responsible for the smearing of sharp corners. On the other hand, the reconstruction technique plays a major role; the more continuous is the interface, the better results we get. The errors in the shearing flow test are shown in Table 5 which shows the excellent and best accurate results of our scheme. In addition, the total CPU time analyzed by the improved VOF scheme increases as the velocity fields get more complicated. In this simulation, the relative consumed CPU-time carried out in this work is compared with the results from the edge-matched flux polygons advection (EMFPA), and spline-based interface reconstruction (SIR) method [25], as shown in Table 6. The present improved VOF also gives very compatible CPU time compared to other algorithms.

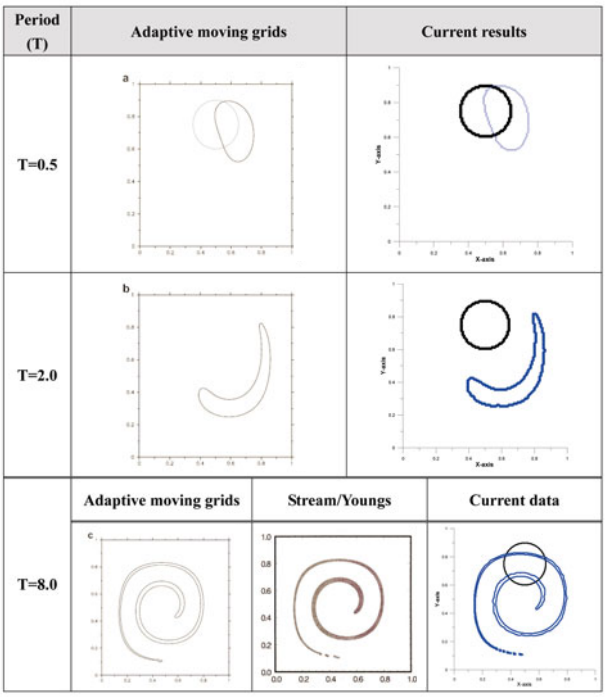


Fig. 12 Reconstructed interface in shearing flow test on uniform grids

Table 5 The errors for the 2D shearing flow test in Rider-Kothe's version ($T = 8$)

Grids	Rider-Kothe	Stream/Puckett	Stream/Youngs	Current data
32^2	4.78e-2	3.72e-2	3.61e-2	2.24e-2
64^2	6.96e-3	6.79e-3	1.00e-2	1.06e-3
128^2	1.44e-3	1.18e-3	2.16e-3	3.92e-4

Table 6 The CPU time-consumed for the Rider-Kothe shearing flow test

Grid Size	EMFPA	EMPFA-SIR	Current data
32^2	3.0	1.0	0.22
64^2	13.6	4.9	1.55
128^2	59.4	23.9	12.52
256^2	248.4	114.1	117.28

4. CONCLUSIONS

An improved PLIC-VOF algorithm for the reconstruction and advection of interfaces has been presented heuristically in the 2D spaces. It is based on a standard PLIC-VOF technique and approximates the interface in each cell described by a segment. The undetermined segment should be drawn from the forward and the inverse relations of cut area corresponding to normal estimations and geometric parameters. A linear mapping expression is employed to accelerate the efficiency of numerical approximations instead of encountering complicated treatments in the programming. We have also investigated the geometric nature of split advection algorithms based on the Lagrangian approach. The performances in 2D spaces have been compared against those of other classical methods for solid body translations and rotations, and for non-uniform vorticity as well. The proposed VOF method demonstrates high efficiency and accuracy through case studies, which reduces drastically the formation of flotsam even for coarse grids, and also compares favorably with other VOF or hybrid VOF methods. Theses encouraging 2D results lead us to the confidence to move to the more challenging complicated 2D and 3D interfacial flow problems, it is underway and will be reported in another paper.

ACKNOWLEDGEMENTS

Financial support from the National Science Council of Taiwan is gratefully acknowledged. It is granted to the National Taiwan University under the Grant No. NSC-99-2221-E-002-089-MY3.

https://doi.org/10.1017/jmech.2012.144 Published online by Cambridge University Press

REFERENCES

1. Chen, S., Johnson, D., Raad, P. and Fadda, D., "The Surface Marker and Micro Cell Method," *International Journal for Numerical Methods in Fluids*, **25**, pp. 749–778 (1997).
2. Torres, D. and Brackbill, J. U., "The Point-Set Method: Front Tracking Without Connectivity," *Journal of Computational Physics*, **165**, pp. 620–644 (2000).
3. Osher, S. and Sethian, J., "Front Propagating with Curvature-Dependent Speed: Algorithm Based on Hamilton-Jacobi Formulations," *Journal of Computational Physics*, **79**, pp. 12–49 (1988).
4. Sethian, J., *Level Set Methods and Fast Marching Methods*. Cambridge University Press (1995).
5. Gueyffier, D., Li, J., Nadim, A., Scardovelli, R. and Zaleski, S., "Volume-of-Fluid Interface Tracking with Smoothed Surface Stress Methods for Three-Dimensional Flows," *Journal of Computational Physics*, **152**, pp. 423–456 (1999).
6. Hirt, C. W. and Nichols, B. D., "Volume of Fluid (VOF) Method for the Dynamics of Free Boundaries," *Journal of Computational Physics*, **39**, pp. 201–225 (1981).
7. Aus der Wiesche, S., "The Cauchy Surface Wave Problem from Viewpoint of a VOF Method," *Computational Mechanics*, **39**, pp. 141–152 (2007).
8. Scardovelli, R. and Zaleski, S., "Note: Analytical Relations Connecting Linear Interfaces and Volume Fractions in Rectangular Grids," *Journal of Computational Physics*, **164**, pp. 228–237 (2000).
9. Sun, D. L. and Tao, W. Q., "A Coupled Volume-of-Fluid and Level Set (VOSET) Method for Computing Incompressible Two-Phase Flows," *International Journal of Heat and Mass Transfer*, **53**, pp. 645–655 (2010).
10. Mencinger, J. and Žun, I., "A PLIC-VOF Method Suited for Adaptive Moving Grids," *Journal of Computational Physics*, **230**, pp. 644–663 (2011).
11. Weymouth, G. D. and Yue, D. K.-P., "Conservative Volume-of-Fluid Method for Free-Surface Simulations on Cartesian-Grids," *Journal of Computational Physics*, **229**, pp. 2853–2865 (2010).
12. Noh, W. F. and Woodward, P., SLIC (simple line interface calculation). *Lecture Notes in Physics*, Springer-Verlag, Berlin, pp. 330–340 (1976).
13. Youngs, D. L., *Time-Dependent Multi-Material Flow with Large Fluid Distortion*, *in Numerical Method for Fluid Dynamics*, edited by K.W. Morton and M.J. Baines, Academic Press, New York (1982).
14. Rider, W. J. and Kothe, D. B., "Reconstructing Volume Tracking," *Journal of Computational Physics*, **141**, pp. 112–152 (1998).
15. Kothe, D. B. and Rider, W. J., "A Comparison of Interface Tracking Methods," *Los Alamos Scientific Laboratory*, Los Alamos, LA-UR-95-1145, New Mexico (1995).
16. Puckett, E. G., "A Volume-of-Fluid Interface Tracking Algorithm with Applications to Computing Shock Wave Refraction," *Proceedings of the Fourth International Symposium on Computational Fluid Dynamics*, pp. 933–938 (1991).
17. Pilliod, J. E. and Puckett, E. G., "Second-Order Accurate Volume-of-Fluid Algorithms for Tracking Material Interfaces," *Journal of Computational Physics*, **199**, pp. 465–502 (2004).
18. Zhang, Q. and Liu, P. L.-F., "A New Interface Tracking Method: The Polygonal Area Mapping Method," *Journal of Computational Physics*, **227**, pp. 4063–4088 (2008).
19. Aulisa, E., Manservisi, S., Scardovelli, R. and Zaleski, S., "Interface Reconstruction with Least-Squares Fit and Split Advection in the Three-Dimensional Cartesian Geometry," *Journal of Computational Physics*, **225**, pp. 2301–2319 (2007).
20. Scardovelli, R. and Zaleski, S., "Interface Reconstruction with Least-Square Fit and Split Eulerian-Lagrangian Advection," *International Journal for Numerical Methods in Fluids*, **41**, pp. 251–274 (2003).
21. Harvie, D. J. E. and Fletcher, D. F., "A New Volume of Fluid Advection Algorithm: The Defined Donating Region Scheme," *International Journal for Numerical Methods in Fluids*, **35**, pp. 151–172 (2001).
22. Ketabdri, M. J., Nobari, M. R. H. and Moradi Laraei, M., "Simulation of Waves Group Propagation and Breaking in Coastal Zone Using a Navier-Stokes Solver with an Improved VOF Free Surface Treatment," *Applied Ocean Research*, **30**, pp. 130–143 (2008).
23. Rudman, M., "Volume Tracking Methods for Interfacial Flow Calculations," *International Journal for Numerical Methods in Fluids*, **24**, pp. 671–691 (1997).
24. Harvie, D. J. E. and Fletcher, D. F., "A New Volume of Fluid Advection Algorithm: The Stream Scheme," *Journal of Computational Physics*, **162**, pp. 1–32 (2000).
25. López, J., Hernádez, J., Gómez, P. and Faura, F., "An Improved PLIC-VOF Method for Tracking Thin Fluid Structures in Incompressible Two-Phase Flows," *Journal of Computational Physics*, **208**, pp. 51–74 (2005).

(Manuscript received May 7, 2012,
accepted for publication August 9, 2012.)

https://doi.org/10.1017/jmech.2012.144 Published online by Cambridge University Press