INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2004; **44**:1–32 (DOI: 10.1002/fld.620)

A computational Lagrangian–Eulerian advection remap for free surface flows

Nasser Ashgriz[1,*,†], Tiberiu Barbat[2] and Gang Wang[3]

[1]*Department of Mechanical and Industrial Engineering, University of Toronto, Toronto, ON, M5S 3G8 Canada*
[2]*Fluent Inc., 220 E. Huron Street, Suite 470, Ann Arbor, MI 48104, U.S.A.*
[3]*Development Department, ANSYS, Inc., 275 Technology Drive, Canonsburg, PA 15317, U.S.A.*

SUMMARY

A VOF-based algorithm for advecting free surfaces and interfaces across a 2-D unstructured grid is presented. This algorithm is based on a combination of a Computational Lagrangian–Eulerian Advection Remap and the Volume of the Fluid method (CLEAR-VOF). A set of geometric tools are used to remap the advected shape of the volume fraction from one cell onto the Eulerian fixed unstructured grid. The geometric remapping is used to compute the fluxes onto a group of neighbouring cells of the mesh. These fluxes are then redistributed and corrected to satisfy the conservation of mass. Here, we present methods for developing identification algorithms for surface cells and incorporating them with CLEAR-VOF. The CLEAR-VOF algorithm is then tested for translation of several geometries. It is also incorporated in a finite element based flow solver and tested in a laminar flow over a broad-crested weir and a turbulent flow over a semi-circular obstacle. Copyright © 2004 John Wiley & Sons, Ltd.

KEY WORDS: free surface flows; volume of fluid method; computational Lagrangian–Eulerian advection remap; laminar flow; turbulent flow

1. INTRODUCTION

Free surface flows refer to flows where there is an interface between a gas and a liquid with a large density difference. Due to a low density, the inertia of the gas is usually negligible, and the only influence of the gas is its pressure acting on the interface. The liquid can thus move freely and the locations of the free surface must be determined as part of the solution process.

One method to model a free surface is to construct a Lagrangian grid that moves with the fluid [1–3]. The Lagrangian method has the advantage of tracking the free surface automatically, but it has problems when the flow field is non-trivially vortical or when the flow has large amplitude surface motions. This results in serious deterioration of the mesh. The

*Correspondence to: N. Ashgriz, Department of Mechanical and Industrial Engineering, University of Toronto, Toronto, Ontario M5S 3G8, Canada.
†E-mail: ashgriz@mie.utoronto.ca

Received 5 December 2002
Revised 11 June 2003

Arbitrary Lagrangian Eulerian (ALE) method is thus introduced with a proper regridding technique [4–9]. Still, these methods cannot successfully handle complex free surface flows involving surface folding and merging. The third method is the Eulerian method, which requires a special way of capturing the free surface of the flow. One of the early methods that was devised to do this is the Marker-and-Cell (MAC) method [10]. The MAC method involves a Lagrangian particle movement and an Eulerian flow field calculation. The marker is used to track the centroid of the fluid element. Its velocity is obtained by averaging the Eulerian velocities in its vicinity. This method, however, suffers from an artificial creation of high or low marker number densities due to the irregularity of the flow field, such as, in converging/diverging flows and stagnation flows. Since the development of the MAC method in 1965, numerous other methods have been developed to model free surface flows and interfaces. Among these are the front tracking method [11] level set method [12–14], the boundary integral methods [15, 16], and methods that approximate interface by line segments and polygons [17, 18].

This paper is concerned with another type of free surface flow modeling method which is based on a concept of a fractional volume of fluid. The volume of fluid (VOF) method is developed to follow the free surface with the volume-tracking feature of the MAC method but without its large memory and CPU costs [19–21]. In this approach, only one storage word, the volume fraction f, is required for each element (or cell). Consider an Eulerian structured grid and an actual curved interface cutting through it. Assume that this curve is the free surface of a liquid domain in a 2-D flow field. Therefore, one can define a volume fraction field in this mesh, f, that can take values between 1 and 0. That is, $f = 0$ represents a cell without liquid ('empty cell'), $f = 1$ a 'full cell,' and $0 < f < 1$ a surface cell, partially filled with liquid, a 'wet cell'. VOF based methods solve the following problem: Given the f field at some moment in time, and the flow field (e.g. u and v components of the velocity field), (i) what is the new f field at the next time step (i.e. the advected volume fraction field), and (ii) what is the shape of the new interface. The evolution of the free surface is computed through the following equation:

$$\frac{\partial f}{\partial t} + u \cdot \nabla f = 0 \tag{1}$$

However, application of this equation to the interface cells is not trivial and requires certain knowledge of the interface shape. An interface can be approximated by a set of line segments. This approximation will converge to the correct interface through mesh refinement. Inside of each cell, this line segment determines uniquely what is the volume fraction inside that cell. However, the reverse correlation is not unique, since more than one location of the interface may determine the same volume fraction in the cell. One has to take into account a group of neighbouring cells in order to reconstruct the interface accurately. The problem of reconstructing an interface by using the volume fraction field data (updated or given at the start) has generated a multitude of approaches and methods. One of the early algorithms is the Piecewise Constant interface method by Noh and Woodward [22], which was referred to as simple line interface calculation (SLIC). Among the other methods are the Piecewise Constant Stair Stepped interface method [19, 23, 24]; Piecewise Linear (PL) interface method [25–28] and Piecewise Second Order interface method [29–33]. In this paper we use a PL VOF based method and all the developments are related to this method. For a general review of the methods used for the numerical simulation of free surface flows see Scardovelli and Zaleski [34].

A popular method for interface advection and reconstruction is the Youngs method [26], which uses a stencil of 3×3 cells in order to fit a line segment inside of the central cell. This method is implemented in the RIPPLE code by Kothe and Mjolsness [35, 36], and further expanded to 3-D flows by Bussmman *et al.* [37]. Ashgriz and Poo [27] developed a method (referred to as FLAIR) based on fitting a line at the common side of two neighbouring cells, such that the liquid fluxes between the two cells are related with the actual slope and location of the interface. A structured grid geometry is embedded in the implementations of these ideas in 2-D and 3-D models such as those in references [38–50]. The resulting algorithms are based on an algebraic identification of the geometric cases of interface location in which the structured pattern of the grid is essential. For example, in FLAIR [27], nine cases are determined by examining the volume fraction field in a pair of surface cells. Although the set can be reduced to case nine after some transformations, this case has also four subcases by surface orientation. The four subcases are identified finally by the two values of the f field in the cells through analytical criteria for the 2-D algorithm. The unstructured grids make it very difficult to use the case-by-case, pure Eulerian approaches.

A second order accurate interface normal method which uses unstructured mesh is developed and tested by Mosso *et al.* [51–53] and Kothe *et al.* [54, 55]. In this method given a velocity field each mesh vertex is moved using a step of forward-Euler, backward-Euler or the trapezoidal rule. The volume fractions of mesh cells are assumed to remain the same during this motion. Next, a line in each new cell is reconstructed using its volume fraction and the volume fractions of its neighbours. Finally, the volume of the cell that is on each side of this line segment is exactly redistributed to the original unmoved mesh cells. On extending the work of Mosso *et al.* [51–53] on unstructured but locally rectangular grids, Shahbazi [56] and Shahbazi *et al.* [57] presented the details of implementation of a second order Eulerian–Lagrangian volume tracking on triangular meshes. The method is similar to the one by Mosso *et al.* [51] and is constructed of three parts: Lagrangian phase, reconstruction phase and remapping phase. Shahbazi *et al.* [57] not only demonstrated the accuracy of the method for flow fields with constant interface topology as reported by Mosso *et al.* [51], they also investigated the behaviour of the method for flow fields including large interfacial stretching and tearing. Ubbink and Issa [58] have developed a new method for capturing of fluid interfaces on meshes of arbitrary topology base on a finite-volume technique. The motion of the interface is tracked by the solution of a scalar transport equation for a phase-indicator field that is discontinuous at the interface and uniform elsewhere. This technique utilizes high resolution discretization schemes to ensure preservation of the sharpness and shape of the interface.

In this paper we present another technique that can represent and evolve a free surface on general unstructured meshes using the volume of fluid method. This technique is similar to that of Mosso *et al.* [51] in terms of the movement of the interface, but different in terms of the redistribution of the advected volume fractions in the neighbouring cell. The idea behind it is to move the fluid portion of an element in a Lagrangian sense, and redistribute the fluid portion locally in the Eulerian fixed mesh. This algorithm is referred to as CLEAR-VOF, where CLEAR stands for computational Lagrangian–Eulerian Advection Remap. This algorithm utilizes exact and robust geometric tools with no special requirement on the mesh topology, the aspect ratio, or the mesh orientation. Here, we shall restrict our discussion to two dimensional planar and axisymmetric problems. This paper is organized as follows: Section 2 describes the fundamental concepts of the CLEAR-VOF advection. Section 3 provides a brief

description of the finite element implementation. The accuracy of the CLEAR-VOF algorithm is verified in Sections 4 and 5. Finally, Section 6 summarizes the major findings of the present work.

2. CLEAR-VOF ALGORITHM

2.1. The structure of the advection algorithm

The CLEAR-VOF algorithm is based on an approach for the computation of the fluxes of fluid originating from a certain element ('home' element) towards each of its 'neighbours'. The idea is to *move in the Lagrangian sense* the fluid portion of an element, find out how much of it remains in the 'home' element, and how much of it passes into each of the 'neighbours'. The fluid portion inside each non-empty element is used to define a *polygon of fluid* in that element as shown in Figure 1(a). If the algorithm starts from a given initial domain of fluid, the polygon of fluid in each element is defined by intersecting (as polygons) the initial domain with each element. If the algorithm starts from a given initial volume fraction distribution, an *interface reconstruction procedure* is used to obtain the polygon of fluid in each partially filled element. If the element is full, the polygon of fluid coincides with the element. The vertices of a polygon of fluid are listed in a counterclockwise (CCW) order as shown in Figure 1(a).

The first stage of CLEAR deals with the motion of the fluid volume fractions for each cell of the mesh. Let us consider the case of a gas–liquid interface for this argument. The interface reconstruction block delivers the shape of the 'wet' part of the cell and the volume fraction corresponding to it. This polygonal shape is set in a Lagrangian motion based on the nodal velocities delivered by a flow solver. The result is a polygonal shape that this volume of liquid takes at the next time step, as depicted in Figure 1(b). *Computational geometry* tools are used to design a robust algorithm, which is capable to detect and construct the intersections

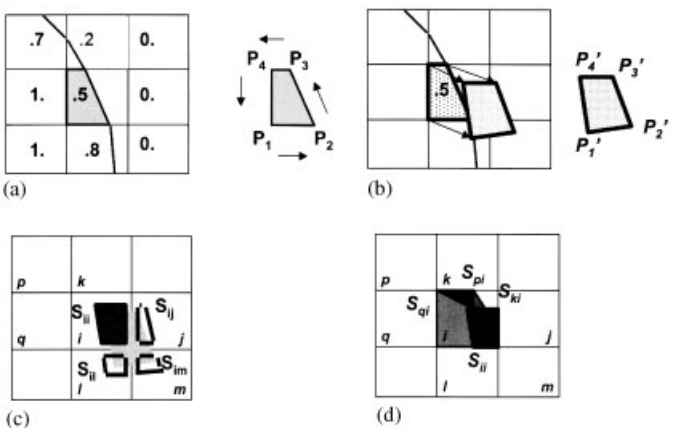

Figure 1. Mechanism of advection in CLEAR-VOF: (a) Initial interface, (b) interface after advection using the node velocities, (c) determination of the volume fraction in each neighbouring cell after the advection and (d) adding the advected volume fractions in each cell.

of the advected liquid domain with the original Eulerian mesh. This will provide the new values for the local f-field, which will be the input for the next stage of the algorithm, as depicted in Figure 1(c).

The second stage sweeps through the cells of the fixed mesh to redistribute and correct the f-field, such that unphysical values are excluded and the mass conservation is assured (Figure 1(d)). This corrected new f-field will serve as input to the interface reconstruction block of the VOF code and to the flow solver for the next time step.

The actual shapes, the number of sides and the angles of the computational cells play no role at all in the formulas contained in the CLEAR algorithm. Generally, the algorithm performs the following operations: (1) Determine the area of a polygon; (2) determine the location of a point with respect to a line segment; (3) determine the intersection of two line segments; (4) determine the location of a point with respect to a polygon and (5) determine the intersection of two polygons. Rigorous mathematical proofs of the theorems used in developing the geometrical tools are presented in computational geometry references [59, 60]. Other results, especially the treatment of the so-called *degenerate* situations, are obtained directly in the process of coding the algorithms.

Once a polygon is identified, it is advected through a Lagrangian local motion using the velocities at its vertices. The vertices of the polygon of fluid are material points in the fluid flow. Each material point undergoes a Lagrangian displacement (ξ, η) which defines the velocity components (u, v):

$$\begin{aligned} u &\equiv \frac{\mathrm{d}\xi}{\mathrm{d}t} \\ v &\equiv \frac{\mathrm{d}\eta}{\mathrm{d}t} \end{aligned} \tag{2}$$

The connection between the Eulerian and the Lagrangian approaches in fluid mechanics is that the velocity field $(u(x, y), v(x, y))$ at some moment is equal to the Lagrangian velocity of the fluid particle which passes through the point (x, y) at that particular instant in time. Then, if the velocity field is solved by some Eulerian solver, Equations (2) can be used to compute the displacements:

$$\begin{aligned} \xi &= \int_t^{t+\delta t} u \,\mathrm{d}t \approx u \cdot \delta t \\ \eta &= \int_t^{t+\delta t} v \,\mathrm{d}t \approx v \cdot \delta t \end{aligned} \tag{3}$$

After the computation of the displacements for each vertex (P_k) of the polygon of fluid, the new locations (P'_k) of these vertices are (see Figure 1(b)):

$$\begin{aligned} x_{P'_k} &= x_{P_k} + \xi_{P_k} \\ y_{P'_k} &= y_{P_k} + \eta_{P_k} \end{aligned} \tag{4}$$

The new polygon of fluid has left the 'home' element. A portion of it remains inside the 'home' element, while several other parts of it cross into the neighbouring elements. Next the amount of fluid volume in each of these portions is determined.

After the new polygon of fluid is defined through its vertices, the fluid volume portions that belong to each of the immediate neighbouring elements are determined using algorithms

developed in computational geometry for intersection of two polygons. In the present code we have implemented an algorithm for computing the intersection of two *convex* polygons. The advected polygons of fluid are restricted to convex shapes through mesh size and time stepping limitations. The algorithm for computing the intersection of two convex polygons gives the resulting polygon through its vertices listed only once in CCW order, along with its area (volume). The following tasks were implemented, using the theoretical basis from References [59] and [60].

1. *Area of a polygon*
   (i) 2-D Cartesian co-ordinates (x, y): We need to compute the areas involved in the definition of the f-field and the ratio of the liquid volume (area in 2-D) to the cell area, regardless of the shapes and orientation of the liquid polygon. Let a polygon (convex or non-convex) P have vertices $v_1, \ldots, v_{n-1}, v_n$, labeled counterclockwise (CCW). Then its area is computed with the following formula:

$$\mathscr{A}(P) = \frac{1}{2} \sum_{i=1}^{n} (x_i y_{i+1} - x_{i+1} y_i) \tag{5}$$

   Each vertex is described by its global Cartesian co-ordinates $v_i = (x_i, y_i)$. The $n+1$ vertex is identified with the first one, $v_{n+1} \equiv v_1$.
   (ii) 3-D axisymmetric co-ordinates (r,z): Let an axisymmetric volume be defined in the plane of co-ordinates (r,z) by a polygonal shape Q described by the 'vertices' $w_1, w_2, \ldots, w_{n-1}, w_n$, labelled CCW. Then the volume enclosed by this axisymmetric surface is given by the following formula:

$$\mathscr{V}(Q) = \frac{\pi}{6} \sum_{i=1}^{n} (r_i + r_{i+1})(r_i z_{i+1} - r_{i+1} z_i) \tag{6}$$

   where each point ('vertex') is described by its (r,z) co-ordinates. Again, the $n+1$ point is identified with the first one, $w_{n+1} \equiv w_1$. This is the only difference between the advection algorithm in 2-D and the advection algorithm in 3-D axisymmetric when approaching the problem with the CLEAR concept. This is a major advantage of our new VOF approach. Previous VOF methods required significant changes to the advection method for the axisymmetric flows (e.g. see Reference [38]).
2. *Location of a point with respect to a line segment*
   Let $p = (x_p, y_p)$ be a given point, and $a = (a_1, a_2) = ((x_{a1}, y_{a1}), (x_{a2}, y_{a2}))$ be a given line segment in the plane. In order to establish the location of the point p with respect to the line segment a, one has to identify which of the following propositions are true: (1) Point p is located strictly on the LEFT half-plane of the line supporting the segment a. (2) Point p is located ON or ALMOST ON the line segment a, but strictly BETWEEN the ends a_1, a_2. The algorithm is based on the following lemma: 'The area of a triangle given by the vertices (v_1, v_2, v_3) is zero if (v_1, v_2, v_3) are collinear, strictly positive if (v_1, v_2, v_3) are listed in CCW order, and strictly negative if (v_1, v_2, v_3) are listed in CW order.'
   We have implemented this algorithm using two logical flags, each one attached with one of the above propositions. The ALMOST option in the second predicate takes care of the floating-point errors, and is implemented using a 'fuzz' factor.

3. *Intersection of two line segments*
   We can distinguish the following situations: (a) Whether the segments have a common point (parallel or not)? (b) Whether this is a proper intersection between two open segments (no ends considered)? In this case, *one* intersection point, (x_i, y_i), is computed. (c) Whether the segments have the same line support? In this case, the intersection is another segment defined through its ends as *two* intersection points. (d) Whether one end of a segment (e.g. a_1) lies on the other segment (b)? In this case, *one* intersection point is returned (a_1).
4. *Location of a point with respect to a polygon*
   The algorithm that we have designed is capable of differentiating the following situations: (a) Point $p=(x_p, y_p)$ is strictly inside the polygon $P=((x_i, y_i),\ i=1,2\ldots n)$. (b) Point p is strictly outside of the polygon. (c) Point p is on or almost on one edge of the polygon. Subroutine also returns on *which* edge is the point located. (d) Point p coincides or almost coincides with one of the vertices. Subroutine also returns the ID of the vertex.
5. *Intersection of two polygons*
   Again we have used the assumption of convex polygons, given as arrays $P_a=(x_i, y_i)$, $i=1,2,\ldots,n_a$ and $P_b=(x_j, y_j)$, $j=1,2\ldots n_b$. The algorithm is based on the method described in Reference [59]. Two polygons of m and n vertices are intersected in at most $2(m+n)$ steps by looking at the intersection of two edges which are 'advanced' along the two boundaries respecting certain rules. We have improved the algorithm such that aside the 'proper' case, the following special situations are correctly interpreted and addressed: (a) The polygons have two parallel edges (the advance mechanism had to be enforced differently). (b) One or more vertices of a polygon are located on or almost on the boundary of the other polygon. (c) One or more vertices of the polygons coincide or almost coincide. (d) $P_b \subseteq P_a$. (e) $P_a \cap P_b = \emptyset$. The code is robust, and returns in all cases the polygon of intersection as an array $P_{int}=((x_k, y_k),\ k=1,2\ldots n_{int})$, with the vertices listed only once, in CCW order and its area.

Once the geometric tools are designed, we can proceed to compute: (i) How much of the advected fluid originating from the home element i, remains in the home element? This is done by intersecting P' with the polygonal element i. Denote this fluid volume by S_{ii} (see Figure 1(c)). (ii) How much of the advected fluid originating from the home element i, has left this element and is now located in the immediate neighbour j? This is done by intersecting P' with the polygonal element j. Denote this fluid volume by S_{ij}. This can be done for all the immediate neighbours of the home element i, as they are listed on the connectivity file. An *immediate* neighbour is an element that has at least one common vertex with the home element, but this can be easily modified to other stencils.

At this moment a very simple check on the conservation of volume can be done by comparing the volume of fluid in the initial polygon of fluid P, to the sum of all fluxes originating from element i, including the S_{ii} portion. A systematic error will occur if the time step is too large, such that the proposed list of immediate neighbours is not covering all the elements touched by the advected polygon of fluid P'. An automatic time stepping mechanism can then be implemented in the algorithm to take care of this problem.

After the advected polygons of fluid originating from all non-empty elements have been redistributed locally in the Eulerian fixed mesh, a sweep through all elements is necessary to update the volume fraction field. Let us consider again a home element i. The new volume

of fluid in it will be the sum of the auto-flux S_{ii} and all the fluxes of the type S_{ki}, originating from the immediate neighbours k.

2.2. Reconstruction method

CLEAR-VOF algorithm uses a piecewise linear (PL) reconstruction method, where the interface is reconstructed as a line segment inside each wet element with $0 < f < 1$. The PL approach to the reconstruction of the interface inside an element is based on considering the interface to be represented by a line segment with the endpoints on the edges of the element. Previous VOF algorithms, designed with the structured mesh geometry embedded in their algebraic relations, made use of very complicated case recognition procedures in order to compute the parameters of the line interface (e.g. see References [27, 38] for interface reconstruction in FLAIR-VOF). None of these methods can be adapted to unstructured mesh environment.

In order to combine the unstructured mesh capability of the CLEAR-VOF with a PL method, we have designed the following structure for the interface reconstruction part of the code: (i) Store the *local* distribution of updated volume fraction field and mesh geometry. Here, '*Local*' means the home element and its immediate neighbours, such as the element i and $j_1, j_2, \ldots, j_9$ in Figure 2. (ii) Compute the unit normal vector $\hat{n}$ to the interface line inside the home element as the unit gradient vector of the volume fraction field in this neighbourhood. (iii) The equation of the line interface in the home element is:

$$g(\vec{x}) = \hat{n} \cdot \vec{x} + c = 0 \tag{7}$$

where $\vec{x} = x\hat{i} + y\hat{j}$. Once the unit normal vector $\hat{n}$ is found, the constant c is computed by requiring the volume fraction of the polygon of fluid delimited by the corresponding line interface to be equal to the given volume fraction for the home element. (iv) For any value of c assumed or computed, the volume fraction inside the home element is determined by constructing the polygon of fluid delineated by the line of Equation (7) inside the home element. Our method will retain for this purpose the vertices of the home element inside the fluid, i.e. the vertices that verify $g(\vec{x}) > 0$, and the intersection points between the line and the edges of the home element.

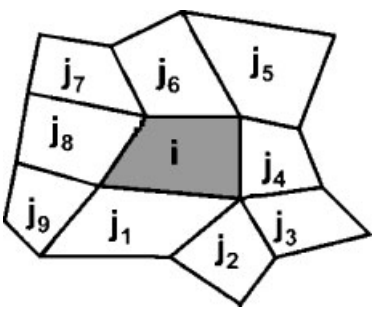

Figure 2. Definition of mesh notations used in CLEAR-VOF.

2.2.1. Computation of the local gradient of the volume fraction field. Several methods are available in the literature for computing the unit vector $\hat{n} \equiv \nabla f / |\nabla f|$; they are reviewed by Kothe *et al.* [54] and Rider and Kothe [33]. For CLEAR-VOF, we have chosen the method of least squares gradient [54]. The method is powerful since it is tied to any mesh topology or dimensionality, being able to handle any unstructured 2-D or 3-D mesh. In this approach, volume fraction Taylor series expansions f_{TS} are formed from the reference element volume fraction to each neighbour of known volume fraction f_k. The sum of the quantities $(f_{\mathrm{TS}} - f_k)^2$ over the list of immediate neighbours is then minimized in the least squares sense. This amounts to solving a 2×2 linear system:

$$\boldsymbol{A}_i \cdot \nabla f_i = \boldsymbol{b}_i \tag{8}$$

for each home element i with j_i immediate neighbours. In Equation (8), the matrix $\boldsymbol{A}_i$ is

$$\boldsymbol{A}_i = \begin{bmatrix} \sum_{k=1}^{j_i} \dfrac{\delta x_{ik}^2}{d_{ik}} & \sum_{k=1}^{j_i} \dfrac{\delta x_{ik} \delta y_{ik}}{d_{ik}} \\ \sum_{k=1}^{j_i} \dfrac{\delta x_{ik} \delta y_{ik}}{d_{ik}} & \sum_{k=1}^{j_i} \dfrac{\delta y_{ik}^2}{d_{ik}} \end{bmatrix} \tag{9}$$

where

$$\begin{aligned} \delta x_{ik} &\equiv x_k^C - x_i^C \\ \delta y_{ik} &\equiv y_k^C - y_i^C \\ d_{ik} &\equiv \sqrt{\delta x_{ik}^2 + \delta y_{ik}^2} \end{aligned}$$

The representative point for each element in computing the gradient of the volume fraction field has been chosen to be the centroid of that polygonal element: $C_i(x_i^C, y_i^C)$.

In Equation (8), the unknowns are the components of the gradient vector of the volume fraction field at the centroid of the home element i:

$$\nabla f_i \equiv \begin{bmatrix} (\nabla f_i)_x \\ (\nabla f_i)_y \end{bmatrix} \tag{10}$$

And the RHS vector is:

$$\boldsymbol{b}_i \equiv \begin{bmatrix} \sum_{k=1}^{j_i} \dfrac{\delta x_{ik} \delta f_{ik}}{d_{ik}} \\ \sum_{k=1}^{j_i} \dfrac{\delta y_{ik} \delta f_{ik}}{d_{ik}} \end{bmatrix} \tag{11}$$

where the quantities δf_{ik} are defined by the local variations of the volume fraction field in the neighbourhood of the home element:

$$\delta f_{ik} \equiv f_k - f_i$$

The solutions of the linear system (8) are obtained with the usual formulas:

$$(\nabla f_i)_x = (b_1 A_{22} - b_2 A_{12})/\Delta$$

$$(\nabla f_i)_y = (b_2 A_{11} - b_1 A_{21})/\Delta$$

$$\Delta = A_{11} A_{22} - A_{12} A_{21}$$

The special case of $\Delta \approx 0$ corresponds to the physical condition of an *almost* constant volume fraction field in the neighbourhood of the home element that we exclude by applying the interface reconstruction procedure only to wet elements ($0 < f < 1$).

2.2.2. Computation of the line constant. The computation of the line constant c in Equation (7) can be complex and time consuming when a case-by-case approach is used, as in the previous VOF methods. The interface reconstruction method that we have chosen for the present CLEAR-VOF implementation makes use of a numerical solution for the equation that imposes the conservation of fluid volume in the home element; that is, the constant c is found by numerically solving

$$\mathscr{A}_i(c) = \mathbf{f}_i^{n+1} \tag{12}$$

where $\mathbf{f}_i^{n+1}$ is the updated volume (area) of fluid in the home element i. An iterative method based on halving the interval $[c_{\min}, c_{\max}]$ is used to solve Equation (12). The limits $c_{\min}$ and $c_{\max}$ are found by allowing the interface line to pass through each of the home element vertices, computing the volume fraction and isolating the extreme cases $f = 0$ and 1. The method converges no matter what the shape of the home element, the geometry of the mesh or the value of the fluid volume.

2.2.3. Intersection of a line with a polygon. When a line constant c is assumed or computed in Equation (12), we are confronted with determining the vertices of the reconstructed polygon of fluid delimited inside the home element by the interface line, and, of course, producing them in the required CCW order. The known data of the problem are the vertices of the home element and the line equation (7). We have designed an algorithm based on computational geometry that delivers the solution to this problem. The algorithm analyses each edge of the home element polygon, deciding if it lies (i) completely inside the fluid, (ii) completely outside the fluid or (iii) crosses the line interface. In the first case, both endpoints of the oriented edge are added to the polygon of fluid list of vertices. In the second case, no point is added to this list. In the third case, the point of intersection is determined and added to the polygon of fluid list after the inside first endpoint of the oriented edge, or before the inside second endpoint of the oriented edge. At the end of the cycle the polygon of fluid in the home element is determined in the CCW order of its vertices.

2.3. f Field adjustment

After the volume fraction is obtained by the VOF advection step, it will be adjusted locally and globally if necessary. Local adjustment is performed to remove unphysical partial elements, whereas global adjustment is performed to enforce the global mass/volume balance.

Consider a bulk of fluid flow along a no-slip wall emptying the elements behind it as time advances. In reality, there exist certain cases where the polygon may have two vertices lie on the no-slip wall during the reconstruction stage. In such cases, there will always be a certain amount of volume left in the home element, which make it practically impossible to empty these near-wall elements. As time advances, the bulk of fluid may leave behind a row of partial elements rather than empty elements. These effects are commonly referred to as 'flotsam' and 'jetsam', where artificial droplets may form and they may never reattach to the core fluid. Hence, local adjustment is performed to eliminate those isolated droplets. A partial element is reset to be empty if it is not adjacent to at least one full element. Similarly, a partial element is reset to be full if its immediate neighbours are all full elements to avoid an isolated partial element inside the fluid bulk.

Further, the global balance of the fluid volume is usually not maintained due to the imperfection of Eulerian velocity field. In our present solution algorithm, the continuity equation is expressed in a Galerkin weak form. As a result, divergence-free condition is not satisfied exactly, and the error is usually in the same order with the discretization error. This error can result in artificial compressibility of the polygon of fluid during the Lagrangian advection step, and thus introduce local and global imbalance in the fluid volume. Even though this error is very small compared to the total fluid volume, it can accumulate exponentially as time advances. Hence an adjustment is necessary to retain the global balance of the fluid volume. The volume fraction of partial elements are increased or decreased proportionally by using the summation of local imbalances:

$$f_p = f_p + \frac{V_{\text{imb}}}{\Sigma_p f_p V_p} f_p \tag{13}$$

where f_p and V_p are respectively the volume fraction and the volume of partial elements. Here, the summation is performed over all partial elements. Further, V_{imb} is the amount of the total volume imbalance, which is the difference between the volume flowing across the external boundary (in–out) and the change of total volume inside the domain. In the above practice, the volume fraction of a nearly full element may be artificially adjusted to an unphysical value greater than one, and thus need to be reset to one. Although this global adjustment for partial elements introduces a numerical diffusion effect, it is believed that the benefit of global conservation of the fluid volume will certainly outweigh this effect. Hence, the global balance of the fluid volume is always checked, and if an imbalance occurs, the volume fraction will be adjusted to enforce the global balance.

3. FINITE ELEMENT IMPLEMENTATION

3.1. Mathematical formulation

In this section we present the implementation of the CLEAR-VOF algorithm in a finite element based flow solver. The partial differential equations governing the time-dependent incompressible flows are the continuity and Navier–Stokes equations. For a Newtonian fluid with constant density (ρ) and constant viscosity (μ), these equations can be expressed in the

following form:

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \tag{14}$$

$$\rho\left(\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y}\right) = -\frac{\partial p}{\partial x} + \mu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) + F_x \tag{15}$$

$$\rho\left(\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y}\right) = -\frac{\partial p}{\partial y} + \mu\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right) + F_y \tag{16}$$

where u, v, p, F_x and F_y are the velocity components in x- and y-directions, pressure and body forces, respectively.

The weighted residual method is then applied to integrate the momentum equations. After integration by parts, the momentum equations can be rewritten as

$$\int_\Omega \left\{ W\rho\left(\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y}\right) + \mu\left(\frac{\partial W}{\partial x}\frac{\partial u}{\partial x} + \frac{\partial W}{\partial y}\frac{\partial u}{\partial y}\right)\right\} \mathrm{d}\Omega$$
$$= -\int_\Omega W\frac{\partial p}{\partial x}\,\mathrm{d}\Omega + \int_\Omega WF_x\,\mathrm{d}\Omega + \int_\Gamma W\mu\left(n_x\frac{\partial u}{\partial x} + n_y\frac{\partial u}{\partial y}\right)\mathrm{d}\Gamma \tag{17}$$

$$\int_\Omega \left\{ W\rho\left(\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y}\right) + \mu\left(\frac{\partial W}{\partial x}\frac{\partial v}{\partial x} + \frac{\partial W}{\partial y}\frac{\partial v}{\partial y}\right)\right\} \mathrm{d}\Omega$$
$$= -\int_\Omega W\frac{\partial p}{\partial y}\,\mathrm{d}\Omega + \int_\Omega WF_y\,\mathrm{d}\Omega + \int_\Gamma W\mu\left(n_x\frac{\partial v}{\partial x} + n_y\frac{\partial v}{\partial y}\right)\mathrm{d}\Gamma \tag{18}$$

where W is the weighting function, Ω is the finite element domain, and Γ is the domain boundary with (n_x, n_y) as its unit normal vector.

3.2. Numerical approximation

A four-noded isoparametric element was used with bilinear shape functions to approximate both the velocities and the pressure. The streamline upwind Petrov–Galerkin (SUPG) weighting function [61] was used for the convection terms, and the traditional Galerkin weighting function for the rest of the terms. The transient term was treated in a lumped mass fashion by a first order backward difference. The resulting term at the current time level was assembled into the diagonal of the coefficient matrix, and those at previous time level make a contribution to the source vector.

On an element basis, the partially discretized momentum equations can be written as

$$a_{ii}^u u_i = -\sum_j^{j\neq i} a_{ij}^u u_j + f_i^u - \int_{\Omega^e} W\left[\frac{\partial p}{\partial x}\right]^e \mathrm{d}\Omega^e \tag{19}$$

$$a_{ii}^v v_i = -\sum_j^{j\neq i} a_{ij}^v v_j + f_i^v - \int_{\Omega^e} W\left[\frac{\partial p}{\partial y}\right]^e \mathrm{d}\Omega^e \tag{20}$$

Here, a_{ij}'s contain the contributions to the element coefficient matrix from convection, diffusion, and transient terms and f_i's contain the contributions to the source term.

The pressure equation can be obtained by applying the Galerkin method of weighted residuals to the continuity equation:

$$\int_\Omega \left(\frac{\partial W}{\partial x} K_i^u \frac{\partial p}{\partial x} + \frac{\partial W}{\partial y} K_i^v \frac{\partial p}{\partial y} \right) \mathrm{d}\Omega = \int_\Omega \left(\hat{u} \frac{\partial W}{\partial x} + \hat{v} \frac{\partial W}{\partial y} \right) \mathrm{d}\Omega - \int_\Gamma W(un_x + vn_y)\, \mathrm{d}\Gamma \tag{21}$$

where

$$\hat{u}_i = \frac{1}{a_{ii}^u} \left(-\sum_j^{j \neq i} a_{ij}^u u_j + f_i^u \right) \tag{22}$$

$$\hat{v}_i = \frac{1}{a_{ii}^v} \left(-\sum_j^{j \neq i} a_{ij}^v v_j + f_i^v \right) \tag{23}$$

$$K_i^u = \frac{1}{a_{ii}^u} \int_\Omega W \, \mathrm{d}\Omega \tag{24}$$

$$K_i^v = \frac{1}{a_{ii}^v} \int_\Omega W \, \mathrm{d}\Omega \tag{25}$$

Because empty elements have no effect on the motion of the fluid, the finite element equations are assembled only for partial and full elements, The contributions of the full elements are treated in the usual manner, whereas those of the partial elements are modified to reflect the absence of fluid in parts of the elements. In the CLEAR-VOF algorithm, partial elements are reconstructed as polygons of fluid inside the home elements. There are two major problems when such a reconstruction is applied to the solution algorithm. First, the reconstructed polygon can have from three to five vertices. The element assembling routines have to be modified to handle the mesh shape changes. Further, there is no guarantee for the continuity of the interface line segments between two neighbouring partial elements. The presence of discontinuity at interface further complicates the solution algorithm. Instead, we adopt a rather simple reconstruction scheme in the solution algorithm in which the nodal co-ordinates of the partial element is modified to reflect the reduction of the fluid volume. The nodes are moved towards the centre of the element so that the reduced element preserves the same shape as the original element, and the ratio between the two is kept to be equal to the volume fraction of the corresponding partial element. The modified nodal co-ordinates are then used to evaluate the integration of the finite element equations over a reduced integration limit. It shall be noted that this modification is only intended for the evaluation of the finite element equations, and the actual spatial co-ordinates of the nodes are not changed.

3.3. Boundary conditions

Boundary conditions for velocity and other degrees of freedom are required for boundary nodes that belong to at least one non-empty (partial or full) element. For boundary nodes

belonging to only empty elements, on the other hand, the prescribed boundary conditions will remain inactive until those nodes are touched by fluid. Finally, boundary conditions are also applied to nodes that belong to at least one empty element and at least one non-empty element. These nodes represent the transition region between the fluid and the void. These nodes are treated with natural boundary conditions for all degrees of freedom except pressure. For the pressure, a constant value is prescribed to model the free surface.

In order to impose proper boundary conditions on the element-based volume fractions, imaginary elements are created along the exterior boundary to act as neighbours to the elements forming the boundary. Two types of boundary conditions are applied on these imaginary elements. The imaginary elements can be specified as either full or empty depending on the imposed volume fraction value. Currently, partial imaginary elements are not allowed on the boundaries. These boundary volume fractions will serve as a neighbour value when determining the unit vector normal to the interface. For the full imaginary elements, a second boundary condition is specified to determine whether the fluid is advected into the computational domain. The boundary is then further identified as either wetting or non-wetting. For the wetting boundary, the imaginary elements have to be full, and the fluid is advected into the domain. For the non-wetting boundary, the fluid or void cannot be advected into the domain.

4. TESTING THE ADVECTION MODEL

The testing was done on several different unstructured meshes generated with ANSYS-5.3. Figure 3 shows translation of a circle of radius 0.15, with the centre at the point (0.3,0.3), and approximated by a regular polygon of 36 points (the domain is 1×1). A quadrilateral mesh (A) with 604 elements was used in this case. We have prescribed a *solenoidal* flow field with $\nabla \cdot \vec{u} = 0$. Translational flows verify the condition that the vorticity field is zero: $\vec{\omega} \equiv \nabla \times \vec{u} = 0$, which in a 2-D system is equivalent to $(\partial v/\partial x) - (\partial u/\partial y) = 0$. The following prescribed flow field is used: $u = 1.0$ and $v = 1.0$. The test used a time step of $\mathrm{d}t = 0.001$. The exact position is shown with thick dotted line for each time frame. The advected circle matches the initial circle very closely up to $t = 0.4$. Figure 4 shows rotation of an ellipse of semiaxes $a = 0.3$ and $b = 0.15$, centred at (0.5,0.5), and approximated by a polygon of 36 points. A quadrilateral mesh (B) with 1799 elements was used in this case. The vorticity field for this flow is $\nabla \times \vec{u} \neq 0$. The following prescribed flow field is used $u = -\omega(y - y_r)$ and $v = \omega(x - x_r)$, with the value for $\omega = \pi$, and the rotation centre at (0.5,0.5). Time step was $\mathrm{d}t = 0.001$. A complete rotation (2000 time steps) was studied. The exact position (determined by the direct computation of the coordinates of the initial points on the interface) is shown with thick dotted line. Again the results are very good up to $t = 1.75$. A comparison between the initial shape and the shape after a complete rotation is shown in Figure 5, showing good accuracy of this technique. We have also tested advection of two different circles in a x-shear flow, where $u = k(y - y_r)$ and $v = 0$. The flow was studied on a quadrilateral mesh (A) with 604 elements. The initial interface was a circle of radius 0.2, with the centre at the point (0.5,0.5), approximated by a regular polygon of 36 points (Figure 6). Figure 7 shows advection of a circle of radius 0.1, with the centre at the point (0.5,0.5) and approximated by a regular polygon of 36 points. A quadrilateral mesh (A) with 604 elements was used in this case as well. The time step was $\delta t = 0.001$ and the constant $k = 1.0$. The results show

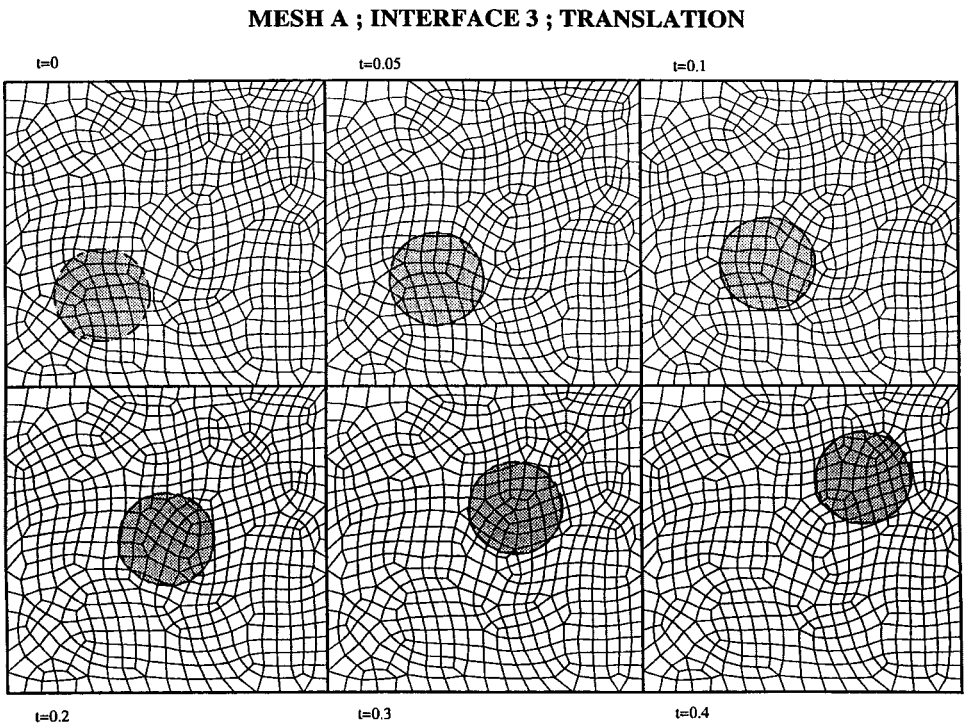

Figure 3. Translation of a circle of radius 0.15 with its centre at $(0.3, 0.3)$ and approximated by a regular polygon of 36 points. Mesh A is a quadrilateral mesh with 604 elements.

that the domain can go through large deformation with interface breakup forming several subdomains.

5. APPLICATION TO FREE SURFACE FLOWS

We now present the numerical results for several problems to illustrate the capability and flexibility of the current finite element implementation. In the following examples, surface tension effects are small and are thus neglected. All calculations are performed on at least two different meshes to assure grid independent results. The calculated results are then compared with experimental and analytical results to verify the accuracy of the current method.

5.1. Laminar flow over a broad-crested weir

Laminar flow over a broad-crested weir is studied here. Figure 8 shows the co-ordinate system, the geometry and the boundary conditions used in the present study. The geometrical and flow parameters selected according to Heinrich and Pepper [62]. The initially stationary fluid is held at a height of $H = 1.5\,\mathrm{m}$, and it is confined by a right vertical wall. The initial pressure field is set to the hydrostatic values of $p_s = \rho g(H - y)$. Here, the fluid density ρ is set to $1\,\mathrm{kg/m^3}$,

and gravity is set to $1\,\mathrm{m/s^2}$. At the time $t = 0$, the right vertical wall is removed, and the fluid falls to the right. The free surface thus evolves in time until a steady state is reached. No-slip wall conditions are specified at the bottom boundary and along the weir surface. At the inlet, the pressure is set to be equal to the hydrostatic pressure p_s, whereas the vertical velocity component is set to zero in order to suppress any spurious mode near the inlet. Further, a zero pressure is applied at the outlet and along the free surface.

Accordingly to the classical open channel flow theory, there are two key factors controlling the shape of the free surface. The first one is the geometry of the weir and the second one is the free fall characteristics at the end of the hydraulic structure. The presence of the weir imposes a local rise in the bed level on the flow. According to the open channel flow theory, the fluid depth (Y) falls as the flow goes over the weir. If the weir height is large enough, the fall would be enough to give critical depth Y_c over the weir. On the other hand, further increasing the weir height does not decrease the depth, and the fall instead remains at the critical depth. For rapidly varied flows, the specific energy (E_s) is a minimum at the critical depth for a fixed amount of discharge per unit of spanwise width (q). For a fixed amount of discharge, on the other hand, the specific energy is a maximum. Assuming a uniform velocity

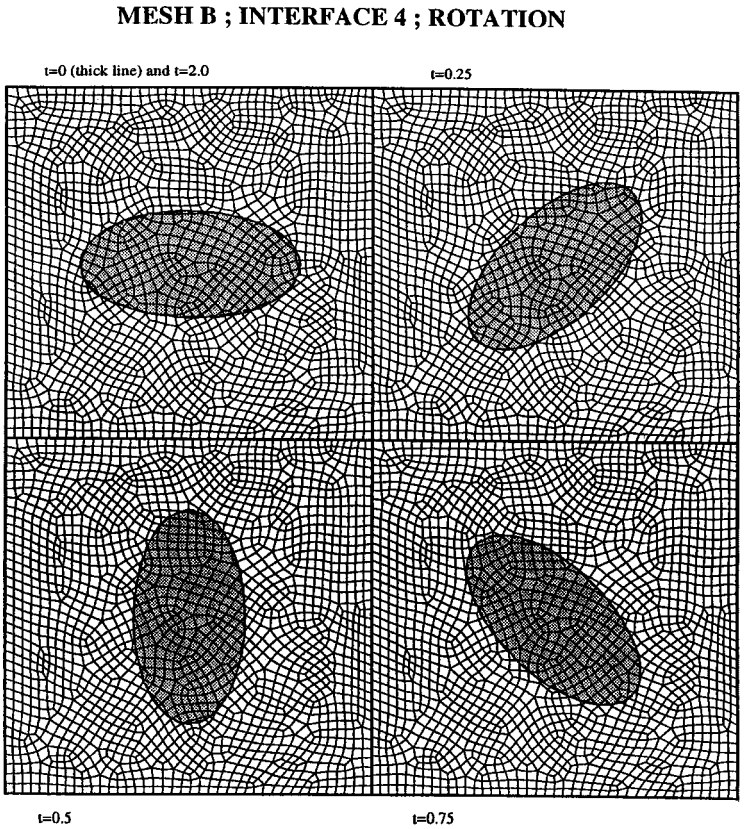

Figure 4. Rotation of an ellipse of semiaxes $a = 0.3$ and $b = 0.15$, centred at $(0.5, 0.5)$ and approximated by a polygon of 36 points. Mesh B is a quadrilateral mesh with 1799 elements.

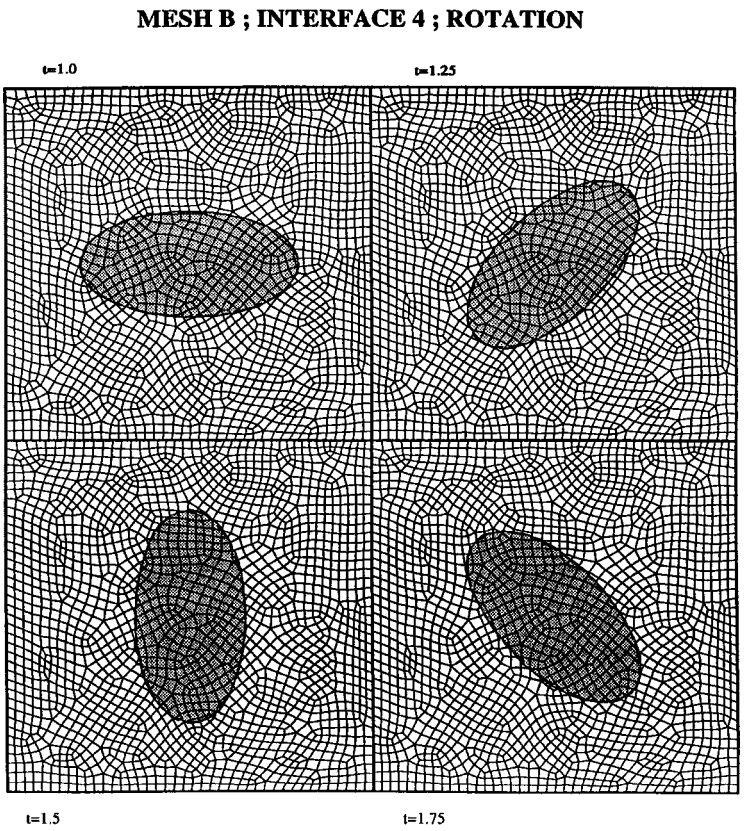

Figure 4. *Continued.*

profile across the depth, then the specific energy equation, with reference to the top surface of the weir, can be expressed in the following form:

$$E_s = Y + \frac{\alpha V^2}{2g} = Y + \frac{\alpha q^2}{2gY^2} \tag{26}$$

where α is the coefficient of energy, and it is used to account for the velocity distributions across a cross-section. In the case of a uniform velocity distribution, α is one, whereas in the case of a non-uniform distribution, it is always greater than one. For turbulent flow in rectangular channels, α is usually less or equal to 1.15. For laminar flows, on the other hand, α can be much higher than one. At the critical depth, the differential of the specific energy is equal to zero, and therefore we can obtain the following equation by assuming that α is constant.

$$\frac{dE_s}{dY} = 0 = 1 - \frac{\alpha q^2}{2gY_c^3} = 1 - \frac{\alpha V_c^2}{gY_c} \tag{27}$$

Hence, the critical velocity V_c can be expressed as $V_c = \sqrt{gY_c/\alpha}$. Substituting this into the specific energy equation, we have $E_c = Y_c + \alpha V_c^2/2g = \frac{3}{2}Y_c$, or $Y_c = \frac{2}{3}E_c$. At the inflow boundary,

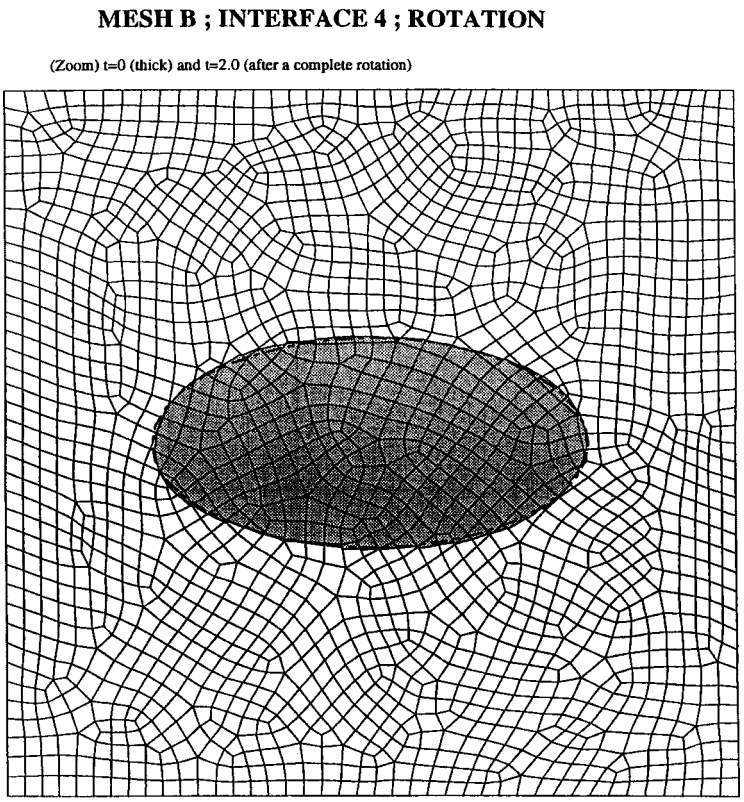

Figure 5. Initial and final shapes of the ellipse described in Figure 4 after one completer rotation.

the velocity head $u^2/2g$ is much smaller than the elevation head H, hence the specific energy can be approximated by the elevation head: $E_c \simeq H$. If the velocity profile is assumed to be uniform over the cross-section, the classical open channel theory predicts a critical depth of approximately $Y_c \simeq 1$ m, a critical velocity of $V_c \simeq 1$ m/s and a discharge of $q \simeq 1$ m^2/s per unit of spanwise width. If the velocity is non-uniform with a known coefficient of energy, then the theory will predict a discharge of $q \simeq 1/\sqrt{\alpha}$ m^2/s per unit of spanwise width.

In the present study, the velocity profile cannot be assumed to be uniform across the depth due to the presence of no-slip walls. The viscous effect is also included in the free surface simulation, and the kinematic viscosity is set to $\mu = 0.01$ m^2/s. The Reynolds number, based on the critical depth Y_c and the critical velocity, is approximately equal to 100.

$$Re = \frac{V_c Y_c}{v} \simeq \frac{1}{v}\sqrt{\frac{8g}{27}}\, H^{3/2} = 100 \tag{28}$$

Systematic grid refinement was performed for this flow. The calculations were performed on three uniformly spaced meshes, consisting of 477, 1908 and 7632 elements, respectively. The mesh sizes are, respectively, given by $\delta x = 0.5$ m and $\delta y = 0.1$ m for the coarsest mesh, $\delta x = 0.25$ m and $\delta y = 0.05$ m for the intermediate mesh, and $\delta x = 0.125$ m and $\delta y = 0.025$ m

for the finest mesh. The time step size is mainly selected such that the home element will not travel across its immediate neighbours. Therefore, each home element will only be able to intersect with its immediate neighbour, and the conservation of volume will be preserved in the advection algorithm. With this in mind, the time step sizes are respectively set to 0.2, 0.1 and 0.05 s. Simulations are carried up to $t = 50$ s to ensure that state is reached.

Figure 9 shows the temporal evolution of the free surface depth Y_b at the exit. The fluid depth falls rapidly in the first two seconds after the right vertical wall is removed, and goes through a moderate change for $t < 20$ s. For $t > 20$ s, the fluid depth undergoes little change, indicating steady state is being reached. Figure 10 provides free surface profiles on the three different meshes at $t = 50$ s. There is a slight difference between the coarsest mesh and the other two meshes, and the difference between the two finer meshes are almost indistinguishable. The fluid depth falls slightly near the inlet region, and falls rapidly as the flow approaches the weir. As the flow passes the leading edge of the weir, the fluid depth continues to fall and further downstream it falls slowly until it approach the outlet where again the drop becomes more substantial. The above observation is also consistent with the numerical study by Heinrich and Pepper [62]. Figure 11 further illustrates the free surface changes by plotting snap shots of the stream functions at five different times ($t = 4$, 8, 12, 16 and 20 s). To better illustrate the flow characteristics, we have selected seven non-uniform

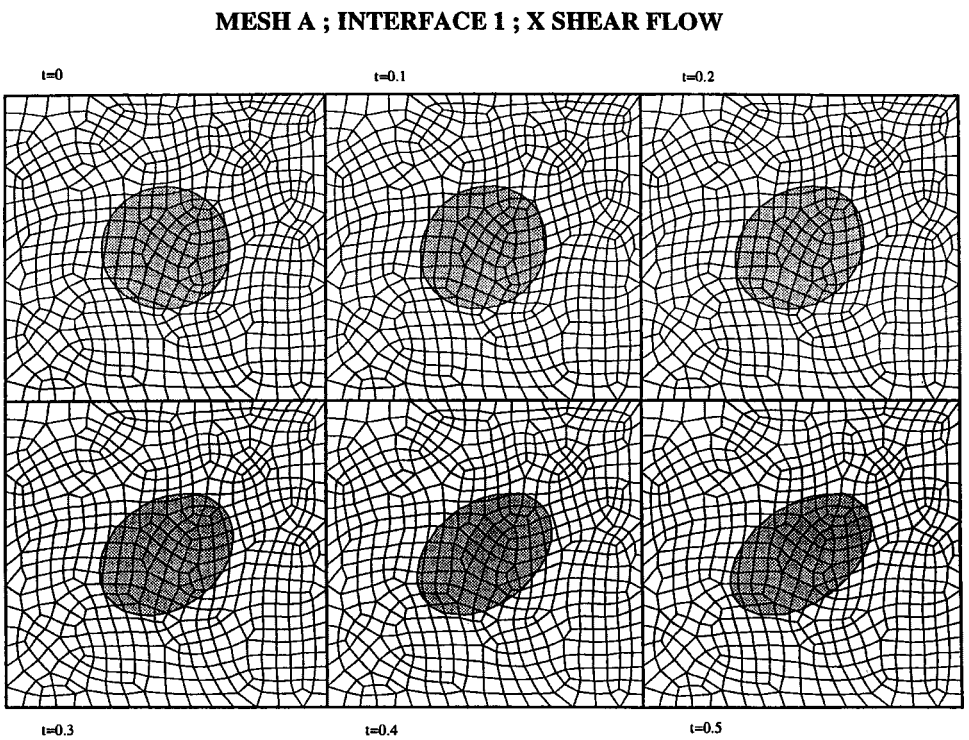

Figure 6. Advection of the circle in a shear flow. The circle radius is 0.2 with its centre at $(0.5, 0.5)$ and approximated by a regular polygon of 36 points. A quadrilateral mesh with 604 elements is used. The shear flow is described by $u = k(y - y_r)$ and $v = 0$.

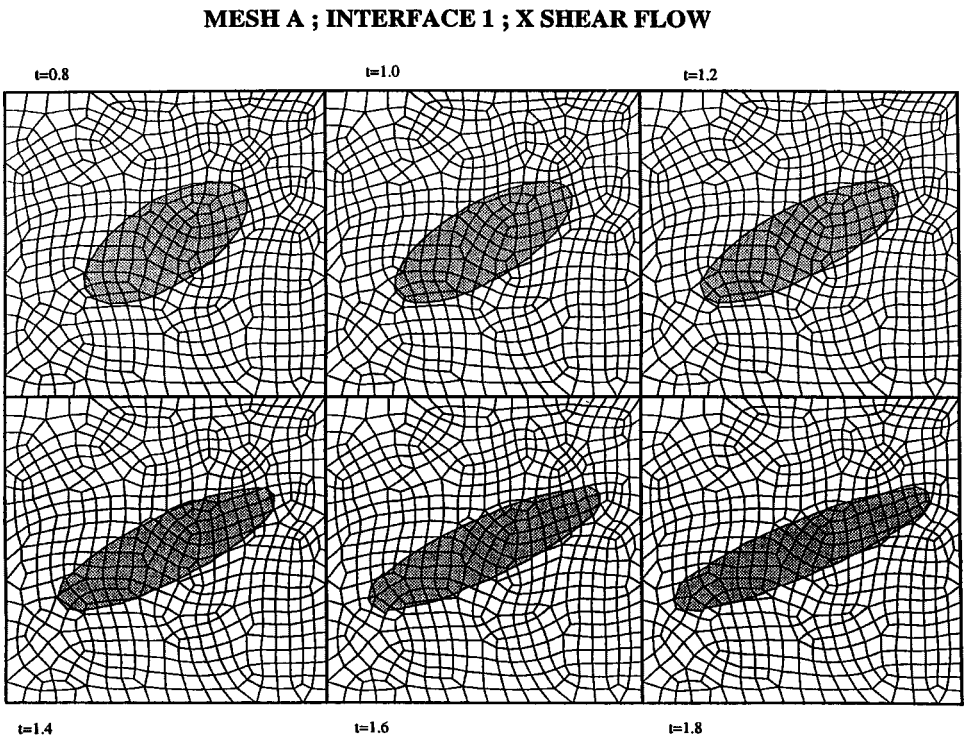

Figure 6. *Continued.*

contour levels of 0.001, 0.01, 0.05, 0.15, 0.3, 0.45 and $0.6\,m^2/s$. The locations of free surfaces are superimposed on the stream function contours for those plots. At $t = 4\,s$, the flow is nearly stagnant in front of the weir, and stream function contours are consistent with the collapse of fluid level near the outlet region. As the time increases, the stream function contours become more affected by the flow from the inlet, and less affected by the initial fluid collapse near the outlet. According to Figure 9, the exit depth is seen to be $Y_b = 0.673\,m$ at the steady state. Further, the present study predicts a discharge of $q_b = 0.845\,m^2/s$ per unit of spanwise width. This corresponds to a exit mean velocity of $V_b = 1.256\,m/s$, and a local depth-based Froude number of $Fr_b = 1.531$. Therefore, the presence of the boundary layer near the weir surfaces gives rise to a non-uniform velocity profile across the flow, as shown in velocity vector plot of Figure 12. As a result, the discharge is approximately 15% lower than the discharge predicted with the assumption of uniform velocity distributions across the depth. On the other hand, if we assume the coefficient of energy is constant, then the present prediction corresponds to $\alpha = 1.40$ according to the classical open channel theory.

Figure 13(a) and 13(b) further presents the velocity profiles along both the inlet and the outlet. Results are plotted on three different meshes to further verify the grid independency. At the inlet, the velocity profiles near the bottom surface are nearly indistinguishable from each other. Near the free surface, on the other hand, large difference are observed between the coarsest mesh and the other two meshes, and the difference is seen as high as 10%. The

difference between the two finer meshes is, however, much small near the free surface, and it is approximately 4%. At the outlet boundary, the differences between various meshes becomes much smaller, and even the coarsest mesh is able to predict the velocity profile within 5% of error. The largest difference still occurs near the free surface, and it is seen to be about 2%. The coefficients of energy can be easily calculated from the velocity profiles given in Figures 13(a) and 13(b), and they are approximately 2.5 at the inlet and 1.1 at the outlet, respectively.

5.2. Turbulent flow over a semi-circular obstacle

In this example, turbulent flow over a semi-circular obstacle is considered [63, 64]. Figure 14 shows the co-ordinate system, the geometry and the boundary conditions used in the present study. The diameter of the semi-circular obstacle is $R = 0.03$ m and the computational domain extends five diameters both upstream and downstream of the obstacle. The fluid is assumed to have a constant density of $\rho = 1\ \mathrm{kg/m^3}$, and a constant kinematic viscosity of $\mu = 10^{-6}\ \mathrm{m^2/s}$. Further, the gravity is set to $g = 9.81\ \mathrm{m/s^2}$. The simulation starts with no initial fluid in the domain, and a wetting boundary condition is set on the inlet with a fixed elevation head of $H = 0.075$ m. This set of geometrical parameters corresponds to a non-dimensional radius of $R/H = 0.4$. The inlet velocity is set to $u_i = 0.32\ \mathrm{m/s}$ corresponding

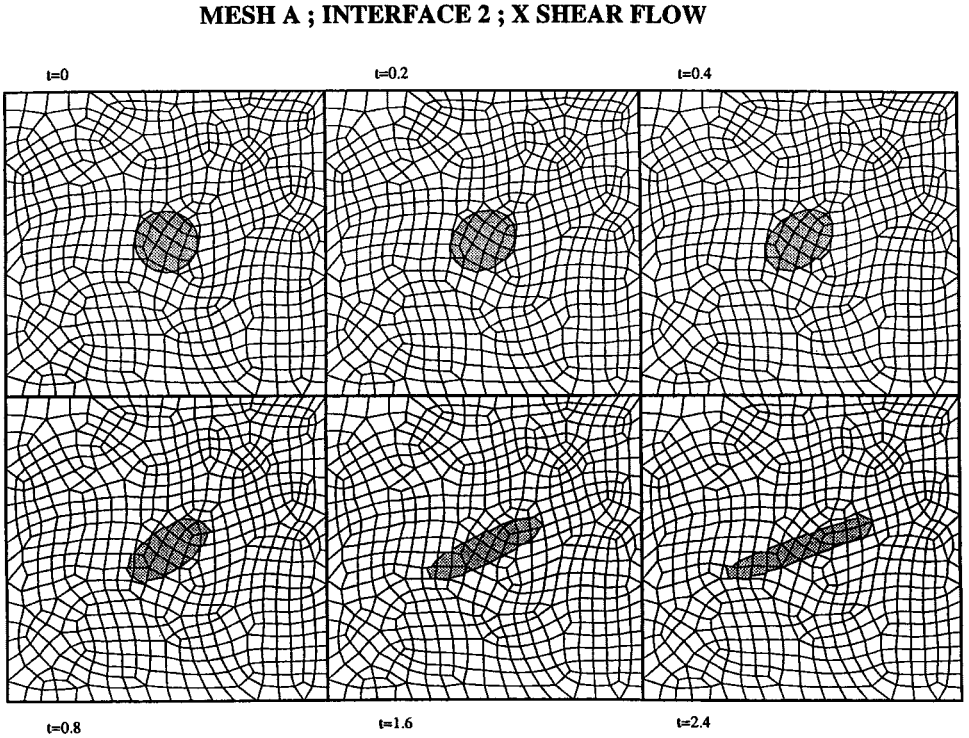

Figure 7. Advection of the circle in a shear flow as described in Figure 6, however, the radius is reduced to 0.1.

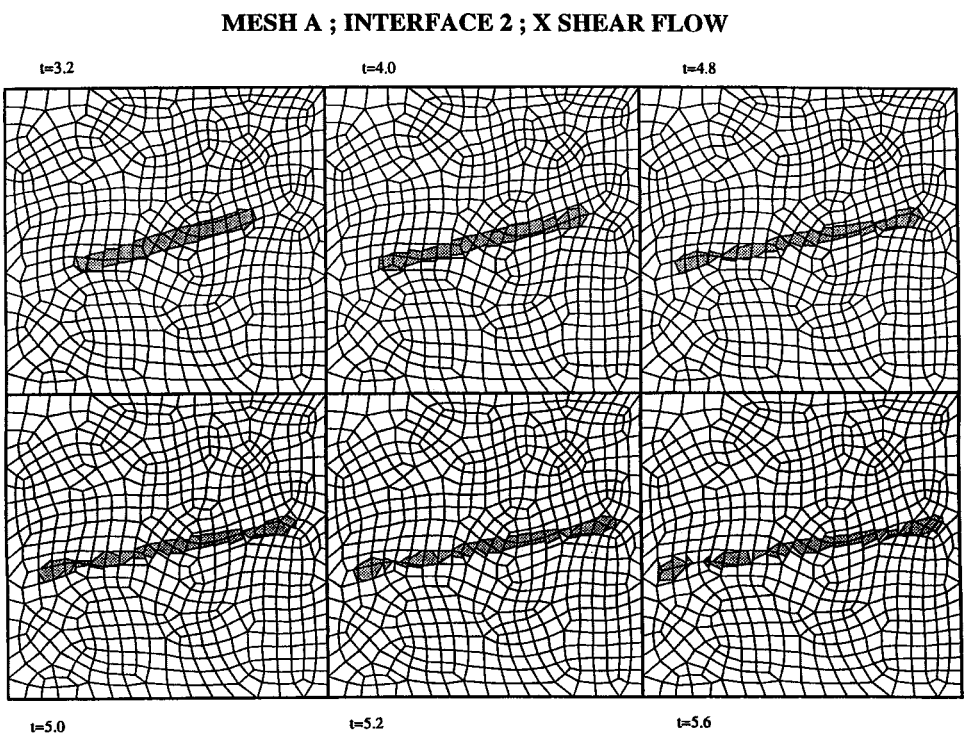

Figure 7. *Continued.*

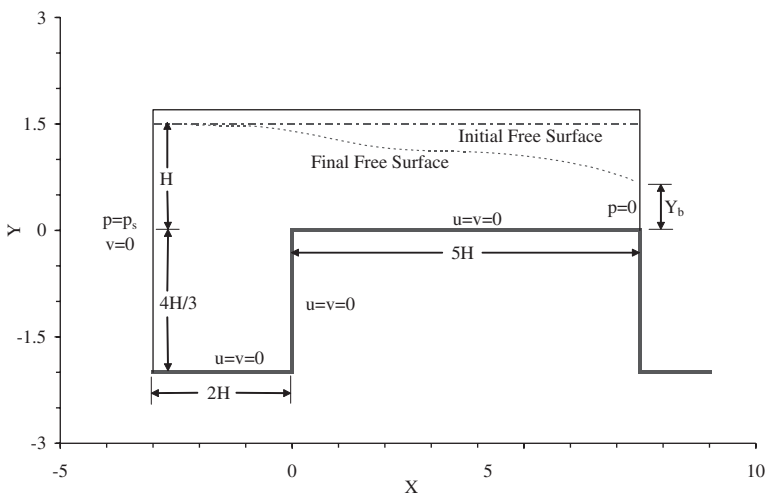

Figure 8. Schematic diagram of the geometry and the boundary conditions used in the laminar flow simulations over a broad-crested weir.

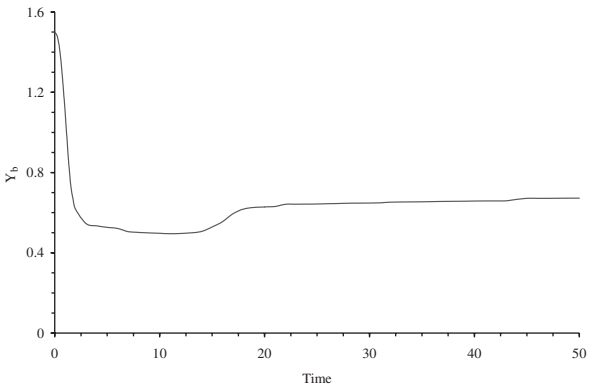

Figure 9. Temporal evolution of the free surface height at the exit of the broad-crest weir.

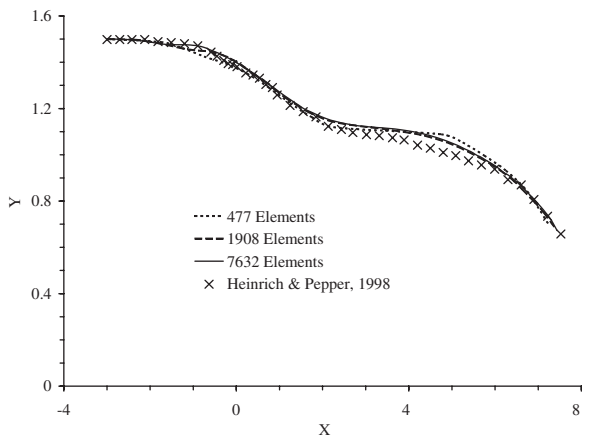

Figure 10. Free surface profile for three different meshes and comparison with those of Heinrich and Pepper [62].

to a Froude number of $F_i = u_i/\sqrt{gH} = 0.373$. On the outlet boundary, natural conditions are specified for the velocity and the pressure is allowed to adjust itself at the outlet during the solution process. Once steady state is reached, the outlet pressure converges to the hydrostatic pressure. The computation is performed using the standard $k-\varepsilon$ model, as it is believed that the turbulence modelling for this case is not important in the prediction of the free-surface shape. The computations are performed on three meshes, consisting of 2320, 9280, 37120 elements, respectively. Figure 15 shows the grid distribution around the semi-circular obstacle on the second mesh. The grid distributions are similar for the other two meshes and they will not be shown here. The time steps are respectively set to 10^{-3} m/s, 5.0×10^{-4} m/s and 2.5×10^{-4} m/s. Simulations are carried up to $t = 10$ s for all three cases to ensure that steady state is reached.

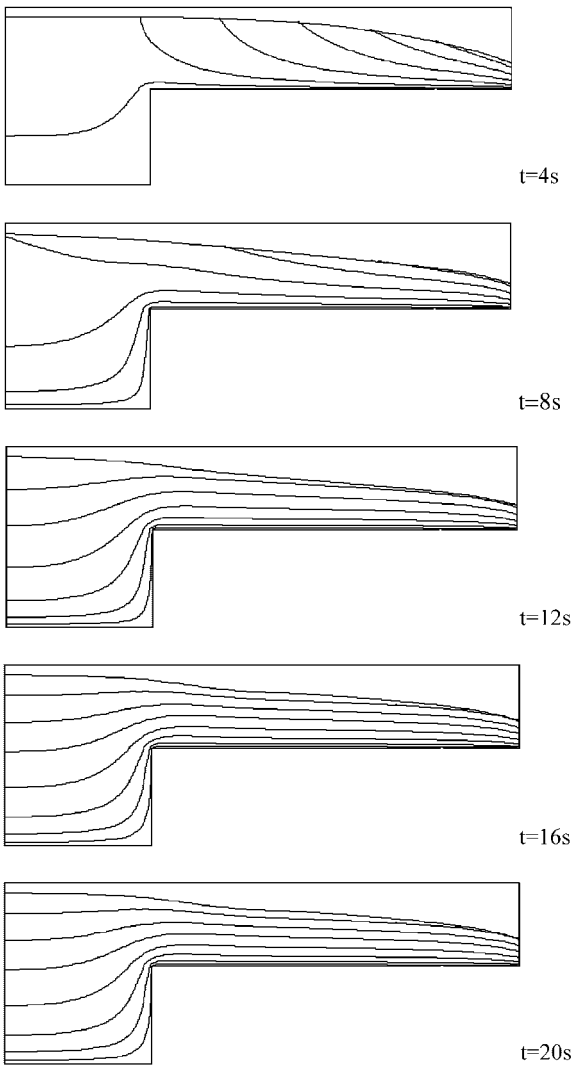

Figure 11. The stream functions at different times for contour levels of 0.001, 0.01, 0.05, 0.15, 0.3, 0.45, and 0.6 m^2/s.

Figure 16 shows the temporal evolution of the free surface depth at two different streamwise locations. The first one is located one diameter upstream of the obstacle ($x/R = -2$), and the other is located at the exit ($x/R = 10$). It is shown that the free surface depth at the exit reaches its asymptotic value much faster than that at the upstream location. This is because of the wave propagation upstream of the obstacle, and as time advances, the magnitude of the wave decreases, and the fluid elevation eventually reaches its steady state level. The time evolution is further illustrated in Figure 17 through several snapshots of the free surface shape

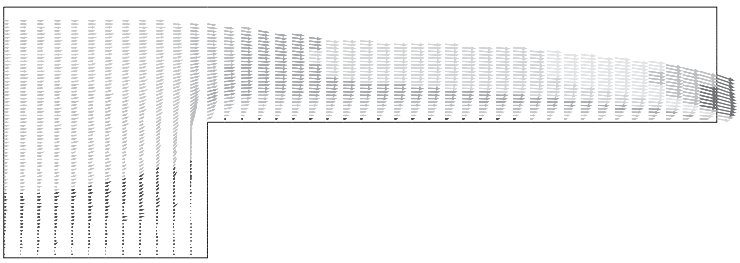

Figure 12. Velocity vector plot at the steady state.

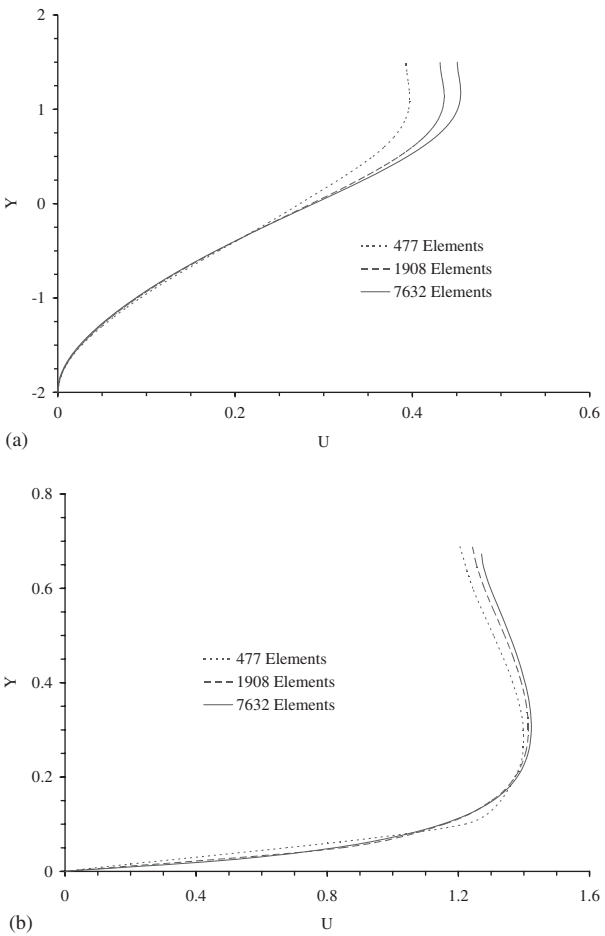

Figure 13. Velocity profile at (a) the inlet and (b) the outlet for three different meshes.

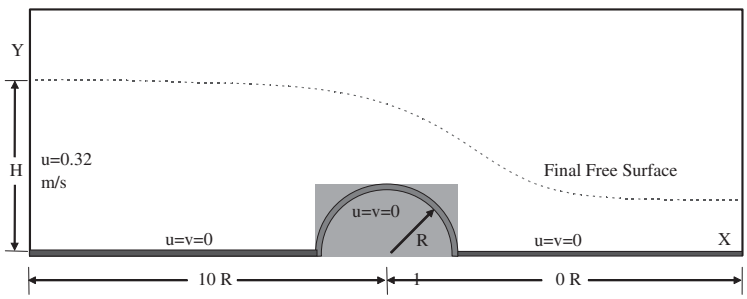

Figure 14. Schematic diagram of the domain used for the turbulent flow over a semi-circular obstacle.

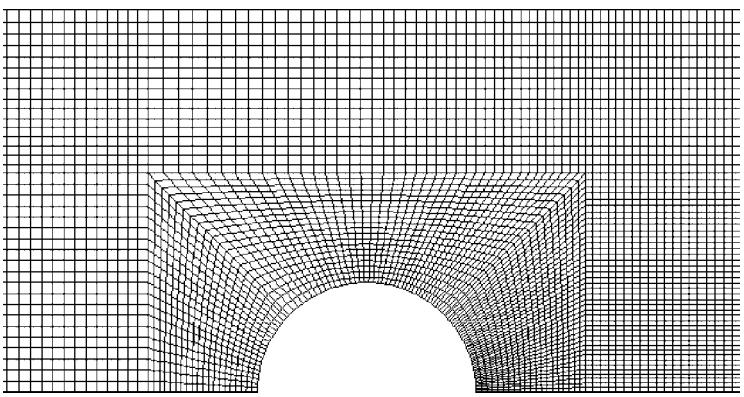

Figure 15. The grid distribution used in the simulation.

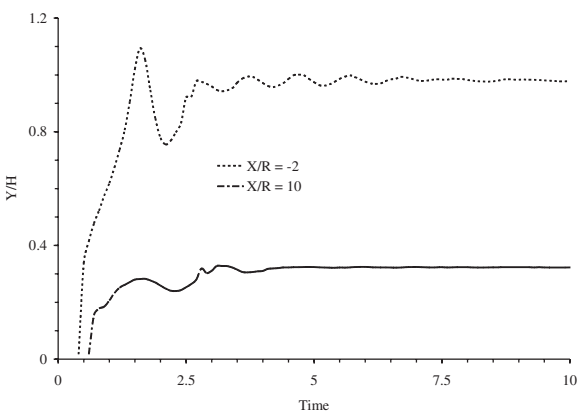

Figure 16. The temporal evolution of the free surface height at two different stream-wise locations for the problem described in Figure 14.

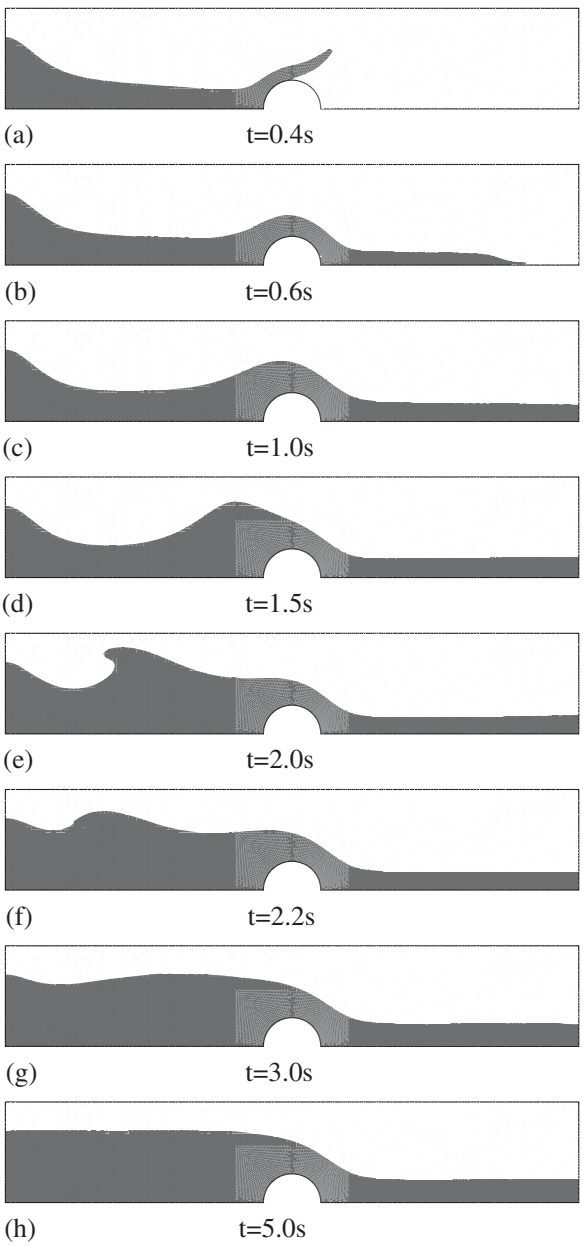

Figure 17. The liquid configuration at different times for Figure 14.

at a series of time instances. For $t>0$, the fluid enters the domain, and it flows along the bottom wall until it hits the semi-circular obstacle. The fluid is then reflected away from the obstacle and becomes a free jet, as shown by Figure 17(a) at $t=0.4$ s. Due to gravity, the free jet eventually falls down and hits the ground splitting into two parts. One part flows forward

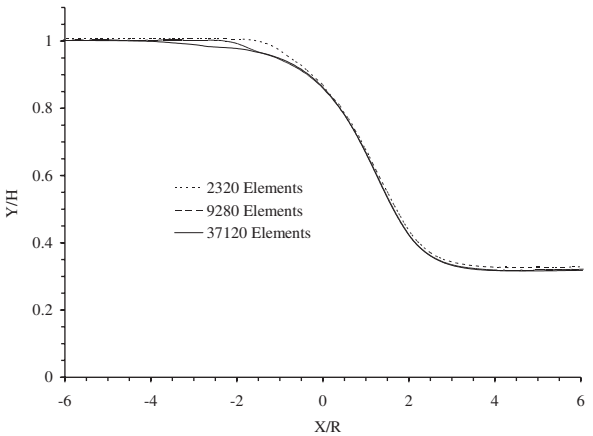

Figure 18. Free surface profile near the obstacle for three different meshes.

and eventually out of the domain, while the other flows backward and eventually fills the cavity between the free jet and the obstacle. At $t = 1.0$ s, a wave is observed just ahead of the obstacle, and it further propagates upstream. As time advances, the upstream fluid elevation increases and the magnitude of the wave decreases. At $t = 5$ s, the flow approaches steady state and the free surface depth downstream of the obstacle is approximately 30% of the inlet level.

Figure 18 further presents the free surface profiles near the obstacle for the three cases. The largest difference is observed to occur at the location about one diameter upstream of the obstacle. The difference between the coarsest mesh and the other two meshes is approximately 4%, and it is less than 2% between the two finer meshes. The predicted non-dimensional downstream flow depths are $Y_b/H = 0.335$, 0.327 and 0.323 on the three meshes, respectively. These predictions agrees well with the value 0.31 given by Forbes [64]. Figure 19 shows the velocity profile along the outlet on the three different meshes. Based on those velocity profiles, the mean velocities at the exit can be calculated, and they are $U_o = 0.933$, 0.968 and 0.985 m/s, respectively. The coefficients of energy are respectively given by 1.08, 1.06 and 1.04 for the three cases. At the inlet, the velocity is assumed to have linear profile at the element near the wall. Hence, the numerical mean velocities are slightly lower than the specified value of $u_i = 0.32$ m/s, and the mean values are, respectively, given by $U_i = 0.307$, 0.314 and 0.317 m/s. These velocities correspond to non-dimensional downstream speeds of $U_r = U_o/U_i = 3.04$, 3.08 and 3.11. Based on the Bernoulli equation, we have the following relation:

$$H + \frac{\alpha U_i^2}{2g} = Y_b + \frac{\alpha U_o^2}{2g} \tag{29}$$

Since $Y_b = H/U_r$, and $Fr_i = U_i/\sqrt{gH}$, we can obtain the following expression for Fr_i.

$$Fr_i = \sqrt{\frac{2}{\alpha U_r(U_r + 1)}} \tag{30}$$

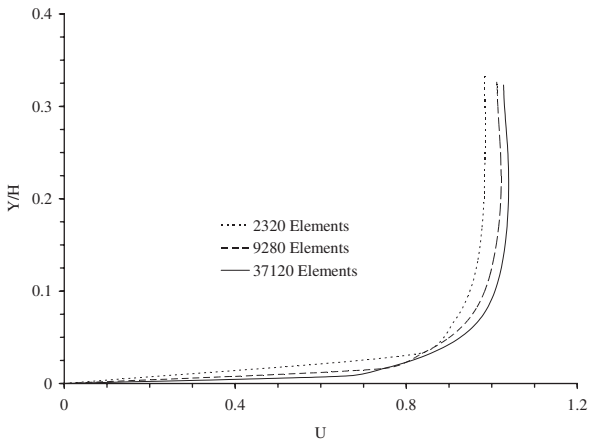

Figure 19. Velocity profile along the outlet for three different meshes.

According to the above equation, the upstream Froude number is approximately 0.388 that is slightly higher than the input setting of 0.37.

6. SUMMARY

A new method of interface advection and reconstruction based on the volume of fluid method (CLEAR-VOF) is described for unstructured meshes. The CLEAR-VOF algorithm consists of two main components. One is the reconstruction of the fluid volume based on the volume fraction field. This reconstruction establishes the shape and location of the free surface. The other component is the advection of the reconstructed polygon of fluid based on a given velocity field. This advection then leads to a new volume fraction field.

The reconstruction of the fluid volume for an element depends on the volume fraction values of both itself and its immediate neighbours. Neighbours here refer to elements sharing a common node. In the reconstruction step, the least squares gradients method is first used to compute the interface normal vector based on the gradient of the volume fraction field in its neighbourhood. The line constant in the interface line equation is obtained by solving an additional equation through halving iterations. This additional equation is formed to enforce the conservation of fluid volume for this element. After obtaining the interface line equation, the vertices of the polygon of fluid are determined by an original algorithm for the intersection of a line with a polygon based on computational geometry theory. In this step, the conservation of fluid volume is satisfied to machine accuracy, and there is no error introduced for the volume fraction field. In summary, the advection of the reconstructed polygon of fluid consists of the following steps: (i) Compute the new locations of the polygon vertices in the Lagrangian displacement step. (ii) Determine the distribution of the advected fluid volume into the neighbourhood using an algorithm for intersection of polygons. (iii) Update the volume fraction at the new time step.

In the Lagrangian step, the polygon of fluid undergoes a Lagrangian movement. The Lagrangian velocity is taken to be the same as the Eulerian velocity at a particular instant in

time. This Lagrangian velocity is then used to calculate the displacements and the new locations of the polygon vertices. This new polygon is used to intersect with the immediate neighbours of the home element in the next step. In the second step, the polygon of fluid at the new time level is then redistributed into its neighbourhood, and no fluid is created or destroyed in this process. Therefore, the volume of fluid in the advected polygon is equal to the sum of all f fluxes originating from this polygon. This conservation of the fluid volume is violated in the following two cases. The first one involves the failure of the polygon intersection algorithm. This occurs when the deformation of the advected polygon is too large during the Lagrangian step such that the convexity of the polygon is lost. The second one involves an incomplete coverage of the advected polygon by the immediate neighbours of the home element. In this case, some f fluxes flow into its far neighbours and are not taken into account by the present algorithm. In either case, the time increment in the Lagrangian step is reduced by half in order to reduce the Lagrangian deformation and the traveling distance of the advected polygon. This automatic reduction in time increment continues until the local balance of fluid volume is preserved. Finally, the f fluxes are regrouped to evaluate the total volume flowing into each home element. Since the volume fraction is just this volume divided by the volume of the home element, this evaluation of volume fraction is exact and there exists no error in this step. Various tests on CLEAR-VOF have shown that this is a powerful technique with great accuracy for solving free surface flows.

ACKNOWLEDGEMENTS

The authors would like to thank the program manager Dale Ostergaard for his constant encouragement and support throughout this development. We would also like to thank Dr Vladimir Zhulin for implementing ANSYS database for VFRC surface loads and Dr Deepak Ganjoo for implementing the postprocessing function and writing the supporting result file.

REFERENCES

1. Ramaswamy B, Kawahara M. Lagrangian finite element analysis applied to viscous free surface fluid flow. *International Journal for Numerical Methods in Fluids* 1987; **7**:953–984.
2. Hansbo P. Lagrangian incompressible flow computations in three dimensions by use for space-time finite elements. *International Journal for Numerical Methods in Fluids* 1995; **20**:989–1001.
3. Muttin F, Coupez T, Bellet M, Chenot JL. Lagrangian finite-element analysis of time-dependent viscous free-surface flow using an automatic remeshing technique: application to metal casting flow. *International Journal for Numerical Methods in Engineering* 1993; **36**:2001–2015.
4. Huerta A, Liu WK. Viscous flow with large free surface motion. *Computer Methods in Applied Mechanics and Engineering* 1988; **69**:277–324.
5. Lewis RW, Navti SE, Taylor C. A mixed Lagrangian–Eulerian approach to modeling fluid flow during mold filling. *International Journal for Numerical Methods in Fluids* 1997; **25**:931–952.
6. Masud A, Hughes TJR. A space-time Galerkin/least-squares finite element formulation of the Navier-Stokes equations for moving domains problems. *Computer Methods in Applied Mechanics and Engineering* 1997; **24**:91–126.
7. Ramaswamy B. Numerical simulation of unsteady viscous free surface flow. *Journal of Computational Physics* 1990; **90**:396–430.
8. Soulaimani A, Fortin M, Dhatt G, Ouellet Y. Finite element simulation of two- and three-dimensional free surface flows. *Computer Methods in Applied Mechanics and Engineering* 1991; **86**:265–296.
9. LeVeque RJ, Shyue K-M. Two-dimensional front tracking based on high resolution wave propagation methods. *Journal of Computational Physics* 1996; **123**:354–368.
10. Harlow FH, Welch JF. Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *Physics of Fluids* 1965; **8**:2182.
11. Osher S, Sethian JA. Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics* 1988; **79**(1):12–49.

12. Adalsteinsson D, Sethan JA. An overview of level set methods for etching, deposition, and lithography development. *IEEE Transactions On Semiconductor Manufacturing* 1997; **10**(1):167–184.
13. Sethian JA. Tracking interfaces with level set. *American Scientist* 1997; **85**(3):254–263.
14. Oguz HN, Prosperitti A. Dynamics of Bubble and Detachment from a Needle, *Journal of Fluid Mechanics* 1993; **257**:111–145.
15. Strain J. A boundary integral approach to unstable solidification. *Journal of Computational Physics* 1989; **85**:342–389.
16. Glimm J, McBryan O. A computational model for interfaces. *Advances in Applied Mathematics* 1985; **6**:422–435.
17. Unverdi SO, Tryggvason G. Computations of multi-fluid Flows. *Physica D* 1992; **60**:70–83.
18. Hirt CW, Nichols BD. Volume of Fluid (VOF) Method for the dynamics of free boundaries. *Journal of Computational Physics* 1981; **39**:201–225.
19. Nichols BD, Hirt CW, Hotchkiss RS. SOLA-VOF: A solution algorithm for transient fluid flow with multiple free boundaries. *Technical Report LA-8355*, Los Alamos National Laboratory, 1980.
20. Torrey MD, Cloutman LD, Mjolsness RC, Hirt CW. NASA-VOF2D: A computer program for incompressible flows with free surfaces. *Technical Report LA-10612-MS*, Los Alamos National Laboratory, December 1985.
21. Torrey MD, Mjolsness RC, Stein LR. NASA-VOF3D: A three-dimensional computer program for incompressible flows with free surfaces. *Technical Report LA-11009-MS*, Los Alamos National Laboratory, July 1987.
22. Noh WF, Woodward PR. *SLIC* (Simple Line Interface Method). *Lecture Notes in Physics* 1976; **59**:330–340.
23. Chorin AJ. Flame advection and propagation algorithms. *Journal of Computational Physics* 1980; **35**:1–11.
24. Barr PK, Ashurst WT. An interface scheme for turbulent flame propagation. *Technical Report SAND82-8773*, Sandia National Laboratory, 1984.
25. DeBar R. Fundamentals of the KRAKEN code. *Technical Report UCIR-760*, Lawrence Livermore National Laboratory, 1974.
26. Youngs DL. Time-dependent multi-material flow with large fluid distortion. In *Numerical Methods for Fluid Dynamics*. Morton KW, Baines MJ (eds), 1982; 273–285.
27. Ashgriz N, Poo JY. FLAIR: flux line-segment model for advection and interface reconstruction. *Journal of Computational Physics* 1991; **93**:449–468.
28. Henderson LF, Colella P, Puckett EG. On the refraction of shock waves at a slow-fast gas interface. *Journal of Fluid Mechanics* 1991; **224**:1–27.
29. Puckett EG, Almgren AS, Bell JB. Marcus DL, Rider WJ. A high-order projection method for tracking fluid interfaces in variable density incompressible flows. *Journal of Computational Physics* 1997; **130**:269–282.
30. Kim SO, No HC. Second-order model for free surface convection and interface reconstruction. *International Journal for Numerical Methods in Fluids* 1998; **26**(1):79.
31. Pilliod JE. An analysis of piecewise linear Interface reconstruction algorithms for volume of fluid methods. *M.S. Thesis*, Department of Mathematics, University of California, Davis, September 1992.
32. Pilliod JE. A second-order unsplit method for modeling flames in two-dimensional compressible flow. *Ph.D. Thesis*, Department of Mathematics, University of California, Davis, September 1996.
33. Rider WJ, Kothe DB. Reconstructing volume tracking. *Journal of Computational Physics* 1998; **141**:112.
34. Scardovelli R, Zaleski J. Direct numerical simulation of free-surface and interface flows. *Annual Review of Fluid Mechanics* 1999; **31**:567–603.
35. Kothe DB, Mjolsness RC, Torrey MD. RIPPLE: A computer program for incompressible flows with free surfaces, Los Alamos National Laboratory Report LA-12007-MS, 1991.
36. Kothe DB, Mjolsness RC. RIPPLE: A new model for incompressible flows with free surfaces. *AIAA Journal* 1992; **30**(11):2694–2700.
37. Bussmann M, Mostaghimi J, Chandra S. On a three-dimensional volume tracking model of droplet impact. *Physics of Fluids* 1999; **11**:1406–1417.
38. Mashayek F, Ashgriz N. Advection of axisymmetric interfaces by the volume-of-fluid method. *International Journal for Numerical Methods in Fluids* 1995; **20**:1337–1361.
39. Mashayek F, Ashgriz N. A height-flux method for simulating free surface flows and interfaces. *International Journal for Numerical Methods in Fluids* 1993; **17**:1035–1054.
40. Mashayek F, Ashgriz N. A hybrid finite element—volume of fluid method for simulating free surface flows and interfaces. *International Journal for Numerical Methods in Fluids* 1995; **20**(10):1363–1380.
41. Mashayek F, Ashgriz N. A spine-flux method for simulating free surface flows. *Journal of Computational Physics* 1995; **122**:367–379.
42. Lafaurie B, Nardone C, Sardovelli R, Zaleski S, Zanetti G. Direct numerical simulation of interface breakup and atomization. In *Proceedings of ICLASS-94*, Rouen, France, July 1994.
43. Zaleski S, Li J, Succi S, Scardovelli R, Zanetti G. Direct numerical simulation of flows with interfaces. In *Proceedings of the 2nd International Conference on Multiphase Flow*, Kyoto, Apr. 3–7, Vol. 2, 1995; PT2-1–PT2-12.
44. Lafaurie B, Nardone C, Sardovelli R, Zaleski S, Zanetti G. Modeling merging and fragmentation in multiphase flows with SURFER. *Journal of Computational Physics* 1994; **113**:134–147.

45. Zaleski S, Li J, Succi S. Two-dimensional Navier-Stokes simulation of deformation and break-up of liquid patches. *Physics Review Letters* 1995; **75**:244.
46. Popinet S, Zaleski S. A front-tracking algorithm for accurate representation of surface tension. *International Journal for Numerical Methods in Fluids* 1999; **30**:775–793.
47. Gueyffier D, Li J, Nadim A, Scardovelli R, Zaleski S. Volume-of-fluid interface tracking with smoothed surface stress methods for three-dimensional flows. *Journal of Computational Physics* 1999; **152**:423–456.
48. Puckett EG. A volume-of-fluid interface tracking algorithm with applications to computing shock wave refraction. In *Proceedings of the 4th International Symposium on Computational Fluid Dynamics*, Dwyer H (ed.), Davis, CA, 1991; pp. 933–938.
49. Rudman M. A volume tracking method for incompressible multifluid flows with large density variations. *International Journal for Numerical Methods in Fluids* 1998; **28**:357–378.
50. Meier M, Yadigaroglu G, Smith B. A novel technique for including surface tension in PLIC-VOF methods. *European Journal of Mechanics B/Fluids* 2002; **21**:61–73.
51. Mosso SJ, Swartz BK, Kothe DB, Ferrel RC. A parallel volume-tracking for unstructured meshes. *Technical Report LA-UR-96-2420*, Los Alamos National Laboratory, Los Alamos, NM 1996.
52. Mosso SJ, Swartz BK, Kothe DB, Clancy SP. Recent enhancement of volume tracking algorithms for irregular grids. *Technical Report LA-CP-96-226*, Los Alamos National Laboratory, Los Alamos, NM, 1996.
53. Mosso SJ, Swartz BK, Kothe DB, Ferrel RC. A parallel volume-tracking algorithm for unstructured meshes. In *Parallel Computational Fluid Dynamics*: *Algorithms and Results Using Advanced Computers*, Capri, Italy, Schiano P, Ecer A, Periaux J, Satofuka N (eds). Elsevier: Amsterdam, 1997; 368–375.
54. Kothe DB Rider WJ, Mosso SJ, Brock JS, Hochstein JI. Volume tracking of interfaces having surface tension in two and three dimensions. *AIAA Paper 96–0859*, 1996.
55. Kothe DB, Williams MW, Lam KL, Korzekwa DR, Tubesing PK, Puckett EG. A second order accurate, linearity-preserving volume tracking algorithm for free surface flows on a 3-D unstructured meshes. *Proceedings of the 3rd ASME/JSME Joint Fluids Engineering Conference*, San Francisco, CA, 1999; July 18–22.
56. Shahbazi K. Remapping volume tracking on triangular meshes. *M.Sc. Thesis*, University of Toronto, June 2002.
57. Shahbazi K, Parachiviou M, Mostaghimi J. Second order accurate volume tracking based on remapping on triangular meshes. *Journal of Computational Physics* 2003; **188**:100–122.
58. Ubbink O, Issa RI. A method for capturing sharp fluid interfaces on arbitrary meshes. *Journal of Computational Physics* 1999; **153**:26–50.
59. O'Rourke J. *Computational Geometry in C*. Cambridge University Press: Cambridge, 1994.
60. Preparata FP, Shamos MI. *Computational Geometry*: *An Introduction*. Springer: New York, 1985.
61. Brooks AN, Hughes JTR. Streamline upwind/Petrov-Galerkin formulations for convective dominated flows with particular emphasis of the incompressible Navier-Stokes equations. *Computer Methods in Applied Mechanics and Engineering* 1982; **32**:199–219.
62. Heinrich JC, Pepper DW. *Intermediate Finite Element Method*: *Fluid Flow and Heat Transfer Applications*. Taylor & Francis: London, 1998; pp. 452–458.
63. Muzaferija S, Peric M. Computation of free-surface flows using the finite-volume method and moving grids. *Numerical Heat Transfer Part B* 1997; **32**:369–384.
64. Forbes LK. Critical free-surface flow over a semi-circular obstruction. *Journal of Engineering Mathematics* 1988; **22**:3–13.

