Available online at www.sciencedirect.com

SCIENCE DIRECT®

Journal of Computational Physics 203 (2005) 668–699

JOURNAL OF COMPUTATIONAL PHYSICS

www.elsevier.com/locate/jcp

The three-dimensional Eulerian–Lagrangian marker and micro cell method for the simulation of free surface flows

Peter E. Raad *, Razvan Bidoae

Department of Mechanical Engineering, Southern Methodist University, Dallas, TX 75275-0337, USA

Received 25 May 2003; received in revised form 9 July 2004; accepted 23 September 2004

Abstract

A new method is presented for the simulation of three-dimensional, incompressible, free surface fluid flow problems. The new technique, the Eulerian–Lagrangian marker and micro cell (ELMMC) method, is capable of simulating incompressible fluid flow problems in Cartesian coordinates where the free surface can undergo severe deformations, including impact with solid boundaries and impact between converging fluid fronts. The method is also capable of handling the breakup of a fluid front from the main body of the fluid as well as their eventual coalescence. The basic solution methodology solves the continuity and the Navier–Stokes equations with a projection scheme and is even able to incorporate a basic k–ε turbulence modeling capability. New approaches are presented for the advection of the free surface, as well as for the calculation of the tentative velocity, final velocity, and pressure fields. The capabilities of the new method are demonstrated by comparing numerical results with experimental studies while the convergence of the new method is demonstrated by spatial and temporal refinement studies.

1. Introduction

Modeling the free surface dynamics of an incompressible fluid represents, in and of itself, a challenging computational problem. The problem is further complicated greatly if wave breaking and multiple fluid bodies are involved and interactions with solid structures are present, as is the case when modeling local tsunami dynamics, wave braking, cavity filling, jet impingement, and the like. Taking the particularly challenging problem of simulating a tsunami wave and its interactions with coastline structures underscores

* Corresponding author. Tel.: +1 214 768 3043/4278; fax: +1 214 768 1473/9900.
E-mail address: praad@smu.edu (P.E. Raad).

0021-9991/$ - see front matter © 2004 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2004.09.013

both the need for a sophisticated computing method and the lack of usefulness of methods with limited ranges of applicability. Specifically, in order to achieve adequate levels of tsunami hazard mitigation, one must be able to predict the effects of a large wave on realistic scenarios of coastal structures. These types of predictions must generally be able to account for the complexities of three-dimensional terrain, river inlets and multiple structures, to name but a few.

Significant progress has been made in numerical modeling, making it possible to simulate large waves and estimate their destructive effects on shoreline structures. Numerical approaches used in the simulation of tsunami waves are very complex because the flow near the runup front exhibits strong nonlinearity in comparison to the motion away from the front, and because of wave breaking near the shore under the influence of sea bottom and shoreline structures. In addition, tsunamis cause substantial erosion and scouring of the shoreline, all of which make it necessary to solve the full physics.

The main difficulty in solving the full Navier–Stokes equations numerically is tied to the free surface boundary. The free surface can advance, brake, and coalesce in time; thus, the algorithm that tracks it must be quite precise and must allow for accurately imposing boundary conditions on the flow. Many methods that track the free surface have been proposed in the past, each with characteristic advantages and disadvantages. The volume of fluid method, level set function, and marker and cell method are examples of primary algorithms for tracking in time the free surface over a fixed grid. The differences between algorithms for tracking the free surface and interfacial flows have been described in significant detail by Floryan and Rasmussen [13], and Scardovelli and Zaleski [37].

The VOF method is one of the most popular schemes for tracking the free surface boundary because of its simplicity. In general, the VOF algorithm solves the problem of updating the volume fraction field C over a fixed grid, given the velocity field u and the field C at the previous time level. In the simplest VOF-methods, the interface is represented with line segments aligned with the grid lines (SLIC method by Noh and Woodward [28]; SOLA-VOF algorithm by Hirt and Nichols [17]); the accuracy of these methods is limited. More accurate VOF methods fit the interface through piecewise linear segments, known as the piecewise linear interface construction (PLIC), with major contributions made by Ashgriz and Poo [3], Rider and Kothe [33], Sabau [36], Rudman [34], Gueyffier et al. [14], and He et al. [16].

The level set method was introduced by Osher and Sethian [29] as a simple and versatile method for computing and analyzing the motion of an interface Γ in two or three spatial dimensions. The interface Γ may bound a multiply connected region Ω and the method is capable of computing the subsequent motion of the interface Γ subject to a velocity field v. Some of the important contributions to the level set method have been made by Munford and Shah [25], Soner and Ambrosio [38], and Ruuth et al. [35].

Many algorithms have been proposed for solving incompressible, free surface, fluid flow problems based on the marker and cell principle. Due to limitations in the free surface tracking schemes employed, the methods developed by Nicholas and Hirt [27], Bush and Phan [6], Ramaswamy and Kawahara [32] could only treat single-valued free surface flows. For a single-valued free surface flow approach, the location of the free surface is a function of time and only one of the spatial coordinates.

The first numerical method capable of treating multi-valued two-dimensional clear fluid flow with a free surface was the marker and cell (MAC) method, proposed in 1965 by Harlow and Welch [15]. They introduced the use of massless markers to track the fluid motion and a novel finite difference solution for the velocity field. Soon thereafter in 1967, Chorin [10] introduced the projection scheme for coupling the equations of motion for an incompressible fluid. Amsden and Harlow [1] subsequently improved and simplified the MAC method and called it the SMAC method. Several modifications were made to the SMAC method in subsequent years. Chan and Street [7] proposed the SUMMAC method that introduced the application of the pressure boundary condition directly at the free surface and the extrapolation of velocity components from the fluid side to obtain velocity boundary conditions. Nakayama and Romeo [26] extended the SMAC method to solve fluid flows that were almost three-dimensional. In 1971, Viecelly [45] proposed the AB-MAC method, a new approach for advancing simultaneous pressure and velocity in one step. A more recent contribution to the MAC method was made by Armenio [2] with SIMAC designed for high Reynolds number free surface problems. Other important contributions include the work of Miyata and Nishimura [24] who developed the TUMMAC and TUMMAC-Vbk methods (the latter being specially designed for wave breaking problems), and the work of Unverdi and Tryggvason [44] who introduced an independent, unstructured grid to describe the interface between immiscible fluids. The method was later extended by Juric and Tryggvason [19] to track the interface resulting from phase change. The method was further developed for multiphase flows by Tryggvason et al. [42].

Tome and McKee [39] proposed the GENSMAC method for calculating free surface flows in two spatial dimensions by solving the full Navier–Stokes equations with a finite difference method. Subsequently, a three-dimensional extension of the method, GENSMAC3D, was proposed in 2001 by Tome et al. [40]. Other major contributions were made by Tsukamoto et al. [43], Kanok-Nukukchai and Tam [20], Lu and He [23], and Popinet and Zaleski [30].

The classical approach for tracking the free surface by the use of marker and cell technique is to establish an initial connectivity law between markers. At each time level, the distance between markers is checked and markers are added or removed. The methodology is time consuming and the algorithm to reconnect the marker list when the free surface breaks or coalesces is very difficult. An innovative idea was proposed by Raad et al. [31] by the introduction of the micro cells and unconnected markers in two spatial dimensions. The method was further developed by Chen et al. [8]. The same concept was later used by Torres and Brackbill [41] in their point set method, which they used to extract the normal vector, the radius of curvature, and the surface from unordered data points residing on the interface.

The new Eulerian–Lagrangian marker and micro cell (ELMMC) method [5] is based on the work of Raad et al. [31], Chen et al. [8,9], Johnson et al. [18], and Fadda et al. [12]. In the ELMMC method, the free surface is tracked by the use of unconnected massless, ‘‘floating’’, Lagrangian markers, named *surface markers*, while the flow field is calculated in a fixed, Eulerian system discretized with rectangular computational cells. The primitive variables are defined on what is referred to as a ‘‘staggered grid’’, in which the velocity components are defined on the cell faces and the scalar variables (i.e., pressure, velocity divergence, kinetic energy, rate of dissipation) are defined on cell centers. The surface markers delineate the full and empty parts of the computational domain and thus make it possible to accordingly flag the computational cells as *full, empty*, or *surface*. Surface cells and their neighboring full cells are subdivided into smaller cells, named *micro cells*. These micro cells, in conjunction with the surface markers, make it possible to prescribe free surface boundary conditions right on the free surface as opposed to at the centers of surface cells. In addition, computation is carried out only in those cells that are flagged as full or surface. The pressure Poisson equation is discretized by the use of a finite difference approach and solved with a preconditioned conjugate gradient method. Global mass conservation is enforced by the use of a mass-imbalance correction scheme.

2. Methodology

In this section, we present the details of the governing equations and boundary conditions, as well as the methodologies used for their respective solutions and implementations.

2.1. Governing equations and projection scheme

The governing equations are the Reynolds averaged Navier–Stokes and continuity equations, which written in Cartesian tensor notation for a Newtonian fluid, appear as:

$$\frac{\partial \bar{u}_i}{\partial t} + \frac{\partial}{\partial x_j}(\bar{u}_i \bar{u}_j + \overline{u'_i u'_j}) = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + v\frac{\partial^2 \bar{u}_i}{\partial x_j \, \partial x_j} + f_i, \tag{1}$$

$$\frac{\partial \bar{u}_i}{\partial x_i} = 0, \tag{2}$$

where $\bar{u}_i$ are the time-averaged velocity components; $\overline{u'_i u'_j}$ are the Reynolds stress components; p is the fluid pressure; ρ is the fluid density; v is the kinematic viscosity; and f_i stands for the external forces.

The system of equations represented by (1) and (2) is solved with a projection method. The first step of the method consists of evaluating the tentative velocity field $(\tilde{u}, \tilde{v}, \tilde{w})$ from the momentum equation by setting the pressure gradient equal to zero. For example, the x-component of the tentative acceleration equation appears as

$$\frac{\partial \tilde{u}}{\partial t} = -\frac{\partial(\bar{u}^2 + \overline{u'u'})}{\partial x} - \frac{\partial(\bar{u}\bar{v} + \overline{u'v'})}{\partial y} - \frac{\partial(\bar{u}\bar{w} + \overline{u'w'})}{\partial z} + v\nabla^2 \bar{u} + f_x. \tag{3}$$

This approach represents an evaluation of the amount of momentum transferred by the fluid flow due to inertia, turbulent kinetic energy, viscous effects, and external forces. The tentative velocity is computed from Eq. (3) according to

$$\tilde{u} \approx \bar{u} + \delta t \frac{\partial \tilde{u}}{\partial t}. \tag{4}$$

The resulting tentative velocity field does not satisfy the continuity equation. Based on the tentative velocity field, an incompressibility deviation function, D, is computed, yielding local measures of the non-satisfaction of the continuity equation for each control volume. The incompressibility deviation function is then used as the non-homogeneous source term in the conventional Poisson pressure potential equation, from which the pressure potential function, $\phi \equiv \frac{1}{\rho}\int p \, \mathrm{d}t$, is computed.

The second step in the projection scheme consists of correcting the tentative velocity field by the use of the gradients of the pressure potential function field, such that

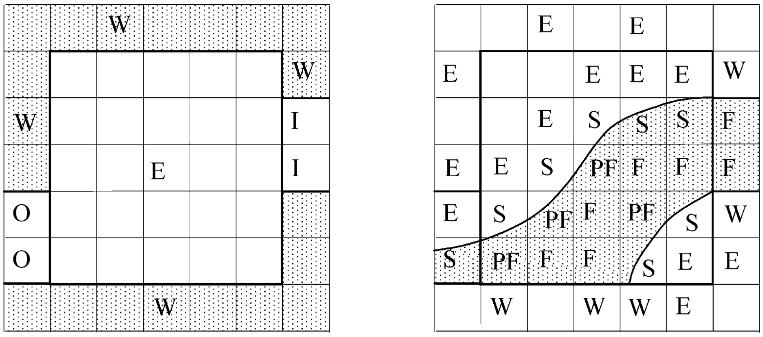
$$\bar{u} = \tilde{u} - \frac{\partial \phi}{\partial x}. \tag{5}$$

This final velocity field obtained after the second step satisfies both the momentum and continuity equations.

2.2. Cell flags

In the ELMMC method, two types of cells are used. By dividing the computational domain with grid planes, a primary type of cells is created, named "*macro cells*". These cells are used in computing the primitive variables of the flow. Following a similar finer division process, a second type of cells is obtained, named "*micro cells*". These cells are used in tracking the free surface and imposing free surface boundary condition. In the ELMMC method, micro cells are obtained by dividing the macro cells into an odd number of parts in each spatial direction, so as to locate a micro cell at the center of a macro cell.

The macro and the micro cells carry two types of flags, namely *geometric* and *computational* flags. The geometric flag (presented on the left side of Fig. 1) is set once at the beginning of a simulation and contains information about the geometry of the domain and the boundary conditions. The computational flag for each macro and micro cell (presented on the right side of Fig. 1) is set at each time level, and contains information on the location of the fluid.



Geometric flags
E – *empty*
W – *wall*
I – *inlet*
O – *outlet*

Computational flags
W – *wall*
E – *empty*
S – *surface*
PF – *premature full*
F – *full*

Fig. 1. Geometric and computational flags.

2.3. Computational cycle

The computational cycle used in the ELMMC method, presented algorithmically in Fig. 2, was designed according to the projection method. Section numbers in the blocks in Fig. 2 refer to the sections in this article where the details are given for each algorithmic step. The first step in the ELMMC method is the marker movement, followed by the computation of the tentative velocity field. Both marker movement and computation of the tentative velocity field are based on information available from the previous time level. The third step is the setting of the computational flags for micro and macro cells based on the new marker positions. For new fluid cells, we assign the external tentative velocity field based on momentum flux exchanged between neighboring cells. The influence of pressure on the change in the velocity field has not been accounted for until this point. Based on the tentative velocity field, the deviation from continuity for each macro cell is calculated and the pressure potential function is computed. The tentative velocity field and the

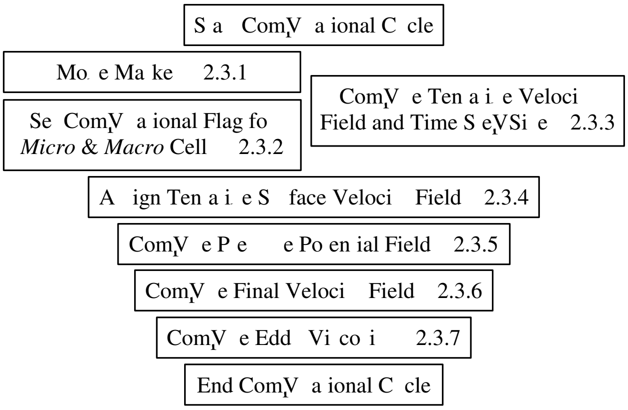

S a ComV a ional C cle
Mo e Ma ke 2.3.1
Se ComV a ional Flag fo
Micro & *Macro* Cell 2.3.2
ComV e Ten a i e Veloci
Field and Time S eVSi e 2.3.3
A ign Ten a i e S face Veloci Field 2.3.4
ComV e P e e Po en ial Field 2.3.5
ComV e Final Veloci Field 2.3.6
ComV e Edd Vi co i 2.3.7
End ComV a ional C cle

Fig. 2. Computational cycle.

pressure potential function are then used to compute the internal velocity field. The external velocity field is corrected such that the continuity equation becomes valid for most surface cells. The last step is the computation of the eddy viscosity, whose values are used to calculate the Reynolds stresses that are used in the next cycle.

2.3.1. Marker movement

The evolution of the fluid interface is accomplished by advecting the *surface* markers from their current to their new locations according to

$$x_k^{(n+1)} = x_k^{(n)} + \bar{u}_k \delta t, \quad y_k^{(n+1)} = y_k^{(n)} + \bar{v}_k \delta t, \quad z_k^{(n+1)} = z_k^{(n)} + \bar{w}_k \delta t, \tag{6}$$

where (n) and $(n+1)$ represent the current and new time levels, respectively; k denotes the marker number; and $\bar{u}_k$, $\bar{v}_k$, and $\bar{w}_k$ represent the velocity components for marker k and are calculated by a modified volume-weighting scheme. For a *surface* marker k, the expressions of local velocities $\bar{u}_k$, $\bar{v}_k$, and $\bar{w}_k$ are

$$\bar{u}_k = \sum_1^8 V_i \bar{u}_i \beta_i / \sum_1^8 V_i \beta_i, \quad \bar{v}_k = \sum_1^8 V_i \bar{v}_i \beta_i / \sum_1^8 V_i \beta_i, \quad \bar{w}_k = \sum_1^8 V_i \bar{w}_i \beta_i / \sum_1^8 V_i \beta_i, \tag{7}$$

where $\bar{u}_i$, $\bar{v}_i$, and $\bar{w}_i$ are the final internal velocities from the preceding computational cycle; V_i are the associated weighting volumes; and β_i are binary switch coefficients (i.e., 0 or 1), used only in special situations to ignore velocity information from outside the fluid. These situations include when the use of velocity information from the entire surrounding would yield incorrect results for the advection of the free surface, such as in the case of two separate fluid fronts impacting each other to be discussed next.

As two fluid fronts approach each other (case on the left-hand side of Fig. 3), a single empty macro cell will eventually separate them. According to the volume-weighting equation, the velocities $\bar{v}_{i,j,k}$, $\bar{v}_{i-1,j,k}$, $\bar{v}_{i+1,j,k}$, $\bar{v}_{i,j-1,k}$, $\bar{v}_{i-1,j-1,k}$, and $\bar{v}_{i+1,j-1,k}$ need to be used in order to calculate the advection velocities for markers A and B. However, velocities $\bar{v}_{i,j,k}$ and $\bar{v}_{i,j-1,k}$ can only be meaningful to one or the other of the two fluid fronts. Consequently, if these values are used to advect both fluid fronts, the convergence of the two fronts will be calculated incorrectly, resulting in both fronts slowing down.

In order to avoid this problem, the ELMMC method uses velocity information only from the internal velocity field for advecting the free surface when different fluid fronts are in close proximity. According to the location of the free surface, for the case presented on the left side of Fig. 3, only velocities $\bar{v}_{i-1,j,k}$ and $\bar{v}_{i-1,j-1,k}$ are used for advecting marker A. Similarly, for advecting of marker B, only the velocities $\bar{v}_{i+1,j,k}$ and $\bar{v}_{i+1,j-1,k}$ are used.

As two fluid fronts continue to approach each other, the distance between the fronts eventually becomes smaller than the dimension of a macro cell. It should be pointed out that in the classical marker and cell

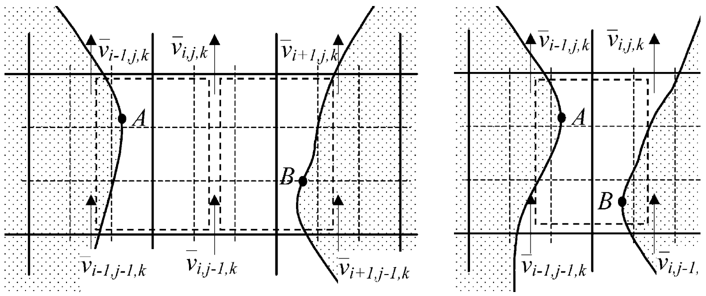


Fig. 3. Converging fluid fronts.

techniques, this case would not exist since the two fluid fronts would have been incorrectly "merged" by now. In contrast, the ELMMC method increases the spatial accuracy by the use of the micro cells for tracking the interface. As long as at least one micro cell exists between the converging fluid fronts, the motion of one front does not affect the advection of the other. The advection of the left front of the case presented on the right-hand side of Fig. 3 is accomplished by the movement of the marker A. The computation of the vertical movement of the marker A involves velocities $\bar{v}_{i,j-1,k}$ and $\bar{v}_{i-1,j,k}$, but $\bar{v}_{i,j-1,k}$ belongs to the other fluid front. Hence, if both velocities were used in the advection of marker A, the impact between the two fluid fronts would be delayed. The correct advection of the left front must be computed based only on velocity $\bar{v}_{i-1,j,k}$. To eliminate $\bar{v}_{i,j-1,k}$ from the computation of the vertical advection of marker A, the switch coefficient, β_i, associated with the velocity $\bar{v}_{i,j-1,k}$ is set equal to zero because neither the micro cell located on the right side nor the one located just under marker A is full. In other words, there is no relevant fluid information path between marker A and marker B.

2.3.2. *Set computational flag for micro and macro cells*

After markers are moved, the flag of each micro cell is set, based solely on the position of the markers. The maximum distance traveled by a marker is initially limited to a distance smaller than a micro cell. Hence, the reflagging is performed only in the cells located in the vicinity of the free surface. After the flags of the micro cells are set, the flag of each macro cell is set based on the flags of its micro cells.

Setting the flag of micro cells is performed in two distinct stages. First, after the advection of the free surface, if the distance between two neighboring markers is greater than an initially prescribed value, a new marker is inserted between them. This process eliminates potential "holes" in the free surface. This algorithm is time consuming (about 15% from the total CPU time) because of the unstructured distribution of markers, but is critical to the integrity of the solution. Second, after an appropriate distribution of markers on the free surface is ensured, the flag of each micro cell is set based on the locations of the markers. Initially, the flag of each micro cell that contains at least one marker is set to *surface* or *wall* type, depending on its geometric flag. Then, a check is performed for each of the new micro cells that carry the *surface* flag, namely if they have at least one neighboring cell of type *empty*. If not, then the computational flag of that micro cell is converted to *full*. The term "neighbor" used here defines any cell that has at least one corner in common with the cell for which a check is being performed. For three-dimensional cases, each cell has 26 neighboring cells. Also the term "*direct* neighbor" will be used later to refer to cells that have at least one face in common with the cell in question. For three-dimensional cases, a cell has six "*direct* neighbor" cells. A similar check is performed for the new cells of type *wall*, but if they do not have a neighbor of type *empty,* then the number of markers inside the new cell is set to zero, and the cell type remains *wall*. The next check is performed again on the new cells that carry the *surface* flag if they have a *direct* neighbor cell that carries an *empty* flag. If not, then the flag of the respective micro cell is converted to the *full* type.

The final part of the flagging step consists of setting the computational flag for macro cells, which is a process based only on the flags of the micro cells. Between two consecutive time levels, the maximum distance that the free surface is allowed to move is much smaller than a macro cell. Thus, as was the case for micro cells, the reflagging procedure for macro cells is performed only for those cells located near the free surface. The algorithm starts by checking the flag of each micro cell contained in those macro cells with a geometrical flag of *empty*. If all micro cell flags are *empty*, then the macro cell flag is obviously set to *empty*. If at least one micro cell carries a *surface* flag, then the macro cell flag is set to *surface* or *premature full*. The flag is set to *surface* if the macro cell has at least one *empty* macro cell; otherwise, it is set to *premature full*. If neither of the above cases is true, then the macro cell flag is set to *full*.

The above procedure is simplified to only two cases for those macro cells that have a geometric flag type of *wall*. In this case, the flags can be set only to either *empty* or *wall*. The flag is set to *wall* for a macro cell when at least one of its micro cells carries a flag of type *wall*.

2.3.3. Computation of the tentative velocity field

The computation of the tentative velocity field is performed for both velocities that are within the fluid (i.e., "internal") and velocities near the free surface but outside the fluid (i.e., "external"). The internal tentative velocity field is computed by the use of the Navier–Stokes equations modified by neglecting the pressure term. The external tentative velocity field is estimated from the best available velocity information with the focus being on ensuring the correct transfer of momentum. The modified Navier–Stokes equations are approximated by the use of the finite volume technique, as opposed to the standard finite difference approach, because of the advantage that the former technique has in correctly estimating the convective momentum flux, not only in the fluid domain but also in the free surface region. The finite difference approach presents the potential advantage of higher order accuracy for flows with a dominant flow direction. However, for problems in which a fluid front impacts another front or a solid obstacle, any potential gains in formal accuracy are negated by the strong likelihood of using erroneous information in the discretization. This is because finite difference approximations assume that the cells are full with a continuous homogeneous fluid.

The ELMMC method employs a complex algorithm to estimate the velocity information on each face of the control volume. For example, if a control volume has at least one micro cell of type *surface* or *empty*, then all connected velocities are checked to see if they belong to the same fluid front. Also, when the free surface undergoes severe deformations, the discretized equation used in computing the momentum flux for one cell can be significantly different from the one used in computing the momentum flux for the neighbors of that cell.

The importance of this level of selectivity and complexity is highlighted with the aid of the example shown in Fig. 4, where $\bar{u}_{i,j,k}$ is an internal velocity, located nearby the free surface but within the fluid front on the right-hand side. If the tentative velocity $\tilde{u}_{i,j,k}$ is computed mechanically in the classical way, the velocities $\bar{v}_{i,j,k}$, $\bar{v}_{i,j-1,k}$, and $\bar{u}_{i-1,j,k}$ would be used. However, the latter velocity belongs to a different fluid front (left), and the first two velocities are computed with information that is extrapolated from that left fluid front. Thus, as was discussed in relation to the correct choice of velocities for advecting markers, if these velocities were used in the estimation of the momentum flux, the resulting momentum flux would be wrong, with one of the overall effects being the incorrect delaying of the impact between the two fluid fronts.

For the computation of the tentative velocity $\tilde{u}_{i,j,k}$ in the current method, the required velocities $\bar{v}_{i,j,k}$, $\bar{v}_{i,j-1,k}$, and $\bar{u}_{i-1,j,k}$ are replaced with values extracted from the fluid front on the right. Based on the shape of the free surface, the required velocity value can be extrapolated from the information in the fluid front on the right or can be computed from the continuity equation, depending on the amount of information available. The same algorithm is used in computing the tentative velocity $\tilde{u}_{i-1,j,k}$, with the

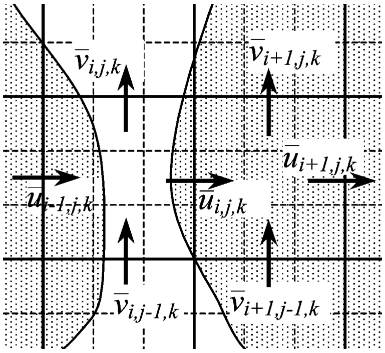


Fig. 4. Internal velocity field near the free surface.

exception that the information from the fluid front on the right is replaced with the velocities from the fluid front on the left. These shifts in the velocity field are done temporarily, such that the initial velocity field remains unchanged. Consequently, the order in which these tentative velocity components are estimated is unimportant.

As previously noted, the appropriate transfer of momentum is the primary physical consideration in the calculation of the tentative velocity field, which is done by the use of the Gauss theorem. Consequently, the momentum flux leaving one control volume is automatically gained by adjacent control volume, and in this way, momentum conservation is guaranteed. However, different procedures are used for internal as opposed to surface velocities, where the law of momentum conservation is not necessarily respected everywhere.

Consider the control volume for the internal velocity $\bar{u}_{i,j,k}$, as presented in the top of Fig. 5. The control volume is projected on the xy and xz planes on the bottom left- and right-hand sides, respectively. The tentative velocity $\tilde{u}_{i,j,k}$ is computed by the use of Eq. (3). The x-momentum flux generated by convective terms through the right face of the control volume $ABCD$ is given by the term $\partial\bar{u}^2/\partial x$, and is calculated with

$$\frac{\partial\bar{u}^2}{\partial x} \approx \frac{T_{\text{right}} - T_{\text{left}}}{\delta x}, \tag{8}$$

where T_{right} and T_{left} represent the rates of x-momentum transferred across the right and left faces of the control volume, respectively. Only the calculation of the x-momentum that passes through the right face $ABCD$ is presented next in detail, but the approach applies to the momentum transfer through the $EFGH$ face.

The x-momentum T_{right} is computed by the use of the relation

$$T_{\text{right}} = \max[U_{\text{cv-r}} \cdot U_{\text{donor}}, 0], \tag{9}$$

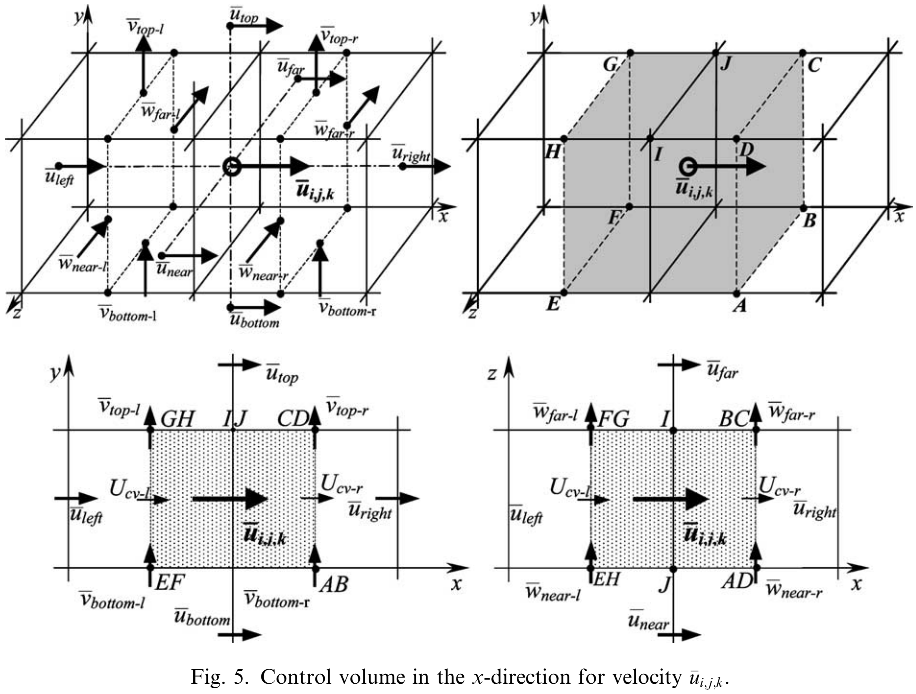


Fig. 5. Control volume in the x-direction for velocity $\bar{u}_{i,j,k}$.

where $U_{\text{cv-r}}$ is the average velocity in the x-direction on face $ABCD$ and U_{donor} is the average velocity of the fluid in the donor control volume. It is possible to obtain negative values for the x-momentum transferred if velocities $\bar{u}_{i,j,k}$ and $\bar{u}_{\text{right}}$ are in opposite directions (which can occur inside the fluid or in the free surface region because of temporary extrapolation of the velocity field). In such cases, T_{right} is intentionally set equal to zero since the momentum transferred in the x-direction should always be positive. The approximation of $U_{\text{cv-r}}$ is given by the equation

$$U_{\text{cv-r}} = \frac{\bar{u}_{\text{right}} + \bar{u}_{i,j,k}}{2}, \tag{10}$$

where the sign of $U_{\text{cv-r}}$ determines the donor control volume, and thus the appropriate value for U_{donor}. If $U_{\text{cv-r}}$ is greater than zero, the fluid flows from the left to the right and thus the donor velocity is $\bar{u}_{i,j,k}$. Otherwise, the fluid flows from right to left, in which case, the donor velocity is $\bar{u}_{\text{right}}$.

In the free surface region, the shape of the free surface will determine the value of $\bar{u}_{\text{right}}$ to be one of the following velocities $\bar{u}_{i+1,j,k},\ \bar{u}_{i,j,k},\ \bar{u}_{i+1,j+1,k},\ \bar{u}_{i+1,j-1,k},\ \bar{u}_{i+1,j,k+1},\ \bar{u}_{i+1,j,k-1}$, or a combination of two or more of them. The same statements are valid for all velocities that are used in the computation of the tentative velocity $\tilde{u}_{i,j,k}$.

Velocity computations near a solid wall represent another situation where the determination of the x-momentum is performed in a different way. The velocity on the right face of the control volume in Fig. 6, $U_{\text{cv-r}}$, is equal to $\bar{u}_{i,j,k}/2$. If the fluid happens to be coming from the right side while fluid is washing down the wall, $\bar{u}_{i,j,k}$ would be negative, and hence U_{donor} should be $\bar{u}_{i+1,j,k}$. But $\bar{u}_{i+1,j,k}$ is equal to zero (solid wall), and thus if the momentum flux is estimated in this straightforward manner, a wrong value would be obtained. Instead, we move U_{donor} away from the wall by a quarter of a cell and we estimate the rate of x-momentum across the right face near a solid wall by the use of the equation:

$$T_{\text{left}} = U_{\text{cv-r}} \cdot U_{\text{donor}} = \frac{\bar{u}_{i,j,k}}{2} \cdot \frac{\bar{u}_{i,j,k}}{4} = \frac{\bar{u}^2_{i,j,k}}{8}. \tag{11}$$

Next, the rate of increase in the x-momentum as a result of the convection across faces $ABEF$ and $CDGH$, which corresponds to the term $\partial\bar{u}\bar{v}/\partial y$, is calculated by the use of the approximation

$$\frac{\partial\bar{u}\bar{v}}{\partial y} \approx \frac{T_{\text{top}} - T_{\text{bottom}}}{\delta y}, \tag{12}$$

where T_{top} and T_{bottom} represent the rates of x-momentum transfer through the *top* and *bottom* faces of the control volume, respectively. Only the calculation of x-momentum that passes through the top faces is presented next in detail.

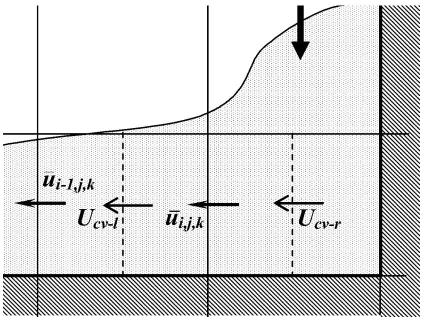


Fig. 6. Computation of the U_{donor} velocity near a solid wall.

The computation of the x-momentum through face *CDGH* is calculated as the sum of the x-momentum fluxes through the two face segments *CDIJ* and *GHIJ*

$$T_{\text{top}} = \frac{\bar{v}_{\text{top-r}} \cdot \bar{u}_{\text{donor-r}} + \bar{v}_{\text{top-l}} \cdot \bar{u}_{\text{donor-l}}}{2}, \tag{13}$$

where $\bar{v}_{\text{top-r}}$ and $\bar{v}_{\text{top-l}}$ represent the average velocities in the y-direction along faces *CDIJ* and *GHIJ*, respectively. The $\bar{u}_{\text{donor-r}}$ and $\bar{u}_{\text{donor-l}}$ represent the average velocities in the x-direction of the appropriate donor. If $\bar{v}_{\text{top-r}}$ is positive, greater than zero, the fluid is flowing from bottom to top through face *BE*, and the donor control volume should be (i + 1,j,k); hence, the donor velocity is $\bar{u}_{i,j,k}$. Otherwise, when the fluid flows from top to bottom, the donor control volume is (i + 1,j + 1,k), and the donor velocity is $\bar{u}_{i,j+1,k}$.

If $\bar{v}_{\text{top-r}}$ and $\bar{v}_{\text{top-l}}$ have the same sign, then a similar result can be obtained by multiplying the average vertical velocity $(\bar{v}_{\text{top-r}} + \bar{v}_{\text{top-r}})/2$ with the donor velocity of the appropriate control volume. But, if $\bar{v}_{\text{top-r}}$ and $\bar{v}_{\text{top-l}}$ have different signs, then the wrong value of the x-momentum would be transferred through face *BC*. For example, if velocity $\bar{v}_{\text{top}-l}$ is equal with $-\bar{v}_{\text{top-r}}$ and T_{top} is computed by the use of an average vertical velocity, then the x-momentum transferred would be zero. However, the x-momentum transferred through face *BC* in ELMMC method is zero only if the horizontal velocities $\bar{u}_{i,j,k}$ and $\bar{u}_{\text{top}}$ are equal.

Next, we calculate the rate of increase in x-momentum as a result of convection across face *BCFG* (and similarly for *ADEH*), which is represented by the term $\partial\bar{u}\bar{w}/\partial z$, by the approximation

$$\frac{\partial(\rho\bar{u}\bar{w})}{\partial z} \approx \frac{T_{\text{far}} - T_{\text{near}}}{\delta z}, \tag{14}$$

where T_{far} and T_{near} represent the rates of x-momentum transfer through the *far* and *near* faces of the control volume, respectively. The calculations of T_{far} and T_{near} are identical to those of T_{top} and T_{bottom}, and are therefore not shown herein.

The viscous contribution to the x-momentum flux is estimated by the use of a modified finite difference approach. This methodology presents the advantage of a correct evaluation of the viscous effects in the free surface region. In the ELMMC method, the variation of the x-momentum inside the control volume is computed with

$$\mu\nabla^2\bar{u} = \mu\left(\frac{\bar{u}_{\text{right}} - 2\bar{u}_{i,j,k} + \bar{u}_{\text{left}}}{\delta x^2} + \frac{\bar{u}_{\text{top}} - 2\bar{u}_{i,j,k} + \bar{u}_{\text{bottom}}}{\delta y^2} + \frac{\bar{u}_{\text{in}} - 2\bar{u}_{i,j,k} + \bar{u}_{\text{out}}}{\delta z^2}\right), \tag{15}$$

where the velocities are as shown in Fig. 5. Away from the free surface region, Eq. (15) reduces to the classical central finite difference formula.

The eddy viscosity influence on the x-momentum flux is computed by the use of a modified finite difference approach. These components are estimated only when the k–ε turbulence model is activated. Based on the Kolmogorov–Prandtl concept, the Reynolds stresses are discretized as follows:

$$\begin{aligned}
-\frac{\partial\overline{u'u'}}{\partial x} &\approx 2\nu\left[\frac{\bar{u}_{\text{right}} - 2\bar{u}_{i,j,k} + \bar{u}_{\text{left}}}{\delta x^2}\right],\\
-\frac{\partial\overline{u'v'}}{\partial y} &\approx \nu\left[\frac{\bar{u}_{\text{top}} - 2\bar{u}_{i,j,k} + \bar{u}_{\text{bottom}}}{\delta y^2} + \frac{(\bar{v}_{\text{top-r}} - \bar{v}_{\text{top-l}}) - (\bar{v}_{\text{bottom-r}} - \bar{v}_{\text{bottom-l}})}{\delta x\,\delta y}\right],\\
-\frac{\partial\overline{u'w'}}{\partial z} &\approx \nu\left[\frac{\bar{u}_{\text{out}} - 2\bar{u}_{i,j,k} + \bar{u}_{\text{in}}}{\delta z^2} + \frac{(\bar{w}_{\text{out-r}} - \bar{w}_{\text{out-l}}) - (\bar{w}_{\text{in-r}} - \bar{w}_{\text{in-l}})}{\delta x\delta z}\right].
\end{aligned} \tag{16}$$

The external forces acting on the control volume are lumped into f_x, the last term on the right-hand side of Eq. (3). In the ELMMC method, f_x represents the free surface tension and the gravitational field. The gravitational field is integrated into the tentative velocity field and its effect is evaluated in the first step of the projection scheme. On the other hand, the surface tension is used to impose the free surface boundary conditions for the potential pressure field [44] and is estimated in Section 2.3.5.

2.3.3.1. Time step size estimation. To optimize the CPU usage, the ELMMC method uses a variable time step computed for each time level based on the flow dynamics. Four distinct criteria are used to establish the time step size; namely, the maximum velocity, tentative acceleration, turbulent kinetic energy, and dissipation of turbulent kinetic energy. The new time step size is evaluated according to the following relations:

$$\mathrm{d}t \leqslant \frac{\delta x_{\mathrm{micro}}}{u_{\mathrm{m}}}, \quad \mathrm{d}t \leqslant \frac{\sqrt{u_{\mathrm{m}}^2 + 2\tilde{a}_x \delta x} - u_{\mathrm{m}}}{\tilde{a}_x}, \tag{17}$$

$$\mathrm{d}t \leqslant \frac{1}{C_{2\varepsilon}}\tau, \quad \mathrm{d}t \leqslant \tau \frac{\tau^*}{|C_{2\varepsilon}\tau^* - C_{1\varepsilon}\tau|}, \tag{18}$$

where $\delta x_{\mathrm{micro}}$ is the dimension of the micro cell; u_m is the maximum tentative velocity in the x direction; $\tilde{a}_x$ is the maximum value of the tentative acceleration field, computed from $\tilde{a}_x = \partial\tilde{u}/\partial t$; $\tau = k/\varepsilon$ is the characteristic decay time; and $\tau^* = k/\Re$ is the characteristic feeding time.

The conditions in Eq. (17) are deduced from the limitation on the maximum distance traveled by a marker during two consecutive time levels, which must be smaller than the dimension of a micro cell. The inequalities in Eq. (18) are due to the k–ε turbulence model and can usually decrease the size of the time step by more than a factor of three; hence they are only computed and used when the model is activated. All four time step inequalities, Eqs. (17) and (18), are calculated in all three directions and the most stringent requirement is used to determine the step size for the next time level.

2.3.4. Tentative surface velocity field

The tentative velocity field in surface cells is assigned and not calculated from the modified Navier–Stokes equations. Also, in order to reduce computational time, the procedure described below is performed only in those cells where the pressure potential function is computed. The tentative surface velocity field is required for estimating the impact pressure boundary conditions, and for computing the incompressibility deviation. The tentative surface velocity field is not computed by the use of the continuity equation, as in other methods. Instead, the tentative surface velocity field is assigned by the use of the best information available such that the momentum conservation law is satisfied in all surface cells. As a result, the incompressibility deviation in surface cells is not necessarily equal to zero.

Two representative cases are shown in Fig. 7, where the tentative surface velocity is being assigned. Dashed lines represent the free surface position before its advection, and solid lines represent the actual position of the free surface. In both cases, the tentative velocity $\tilde{v}_{i,j,k}$ must be assigned. The difference between the two cases presented in Fig. 7 is the state of macro cell (i,j + 1,k); which is *empty* in Fig. 7(a), and *surface* in Fig. 7(b).

In the first case (shown in Fig. 7(a)), in which the tentative surface velocity lies between a *surface* and an *empty* macro cell, the tentative acceleration $\partial\tilde{v}_{i,j-1,k}/\partial t$ is the most appropriate information to use in the computation of the tentative velocity $\tilde{v}_{i,j,k}$. Therefore, the external tentative velocity $\tilde{v}_{i,j,k}$ is set such that

$$\tilde{v}_{i,j,k} = \bar{v}_{i,j,k} + \frac{\partial\tilde{v}_{i,j-1,k}}{\partial t}, \tag{19}$$

where $\bar{v}_{i,j,k}$ is the final velocity computed in the previous time level.

In the second case presented in Fig. 7(b), the tentative surface velocity $\tilde{v}_{i,j,k}$ has two *direct* neighbors ($\tilde{v}_{i+1,j,k}$ and $\tilde{v}_{i,j-1,k}$) that are computed by the use of the simplified Navier–Stokes equations. Thus, a new procedure was developed, based of the main flow direction, to determine which is the most appropriate

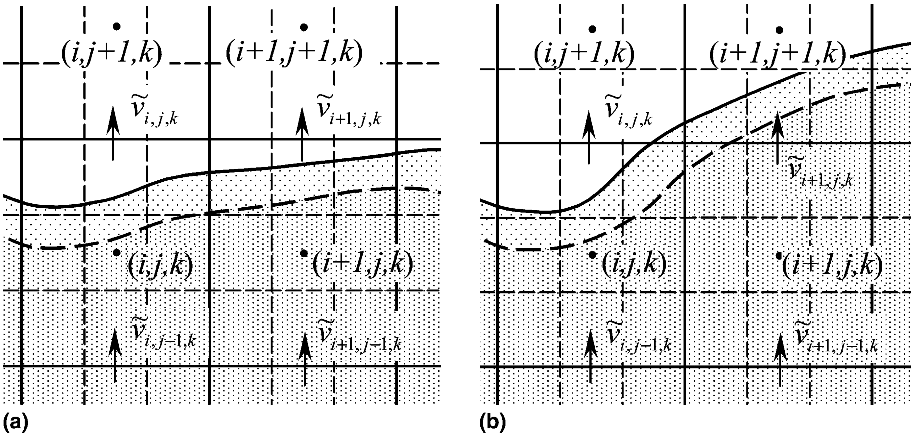


Fig. 7. Assignment of the external tentative velocity field in new fluid cells.

value for the tentative velocity $\tilde{v}_{i,j,k}$. The assignment of the tentative surface velocity is based on the direction of the flow that enters into the control volume for $\bar{v}_{i,j,k}$. The direction of the flow is established by checking the flags of the micro cells within the $v_{i,j,k}$-control volume. For the vertical velocity, the micro cells located on the *left*, *right*, *near* and *far* are examined first. If any of these is a *surface* or a *full* cell, then the fluid must have entered into the $\bar{v}_{i,j,k}$-control volume from that face. Consequently, the tentative surface velocity $\tilde{v}_{i,j,k}$ is assigned to be equal with the tentative velocity value from that direction. If the fluid has entered in the $\bar{v}_{i,j,k}$-control volume from two or more directions, then the tentative surface velocity $\tilde{v}_{i,j,k}$ is computed as an average of their values.

If the previously examined four micro cells are *empty*, then the last two faces on the *top* and the *bottom* are investigated. As before, if any of the micro cells located on one of these faces is a *surface* or a *full* cell, then it is understood that the flow enters in the $\bar{v}_{i,j,k}$-control volume from that direction. However, if *surface* cells are found on both the *top* and *bottom* faces, then the tentative velocity $\tilde{v}_{i,j,k}$ is set equal to the average value of tentative velocities $\tilde{v}_{i,j+1,k}$ and $\tilde{v}_{i,j-1,k}$.

It is possible for all *direct* neighbors to be external velocities. This happens when there is a strip of thin film of fluid extending vertically or a small droplet. In this case, there is no *direct* neighbor that is inside the fluid, the only information that is available is the external forces that act on the control volume, namely the gravitational field. Therefore, the rate of change of momentum is simply approximated from the gravitational acceleration field.

2.3.5. Computation of pressure potential field

After the tentative velocity field has been computed or assigned to the entire computational domain, the incompressibility deviation function is estimated. The Poisson equation governing the pressure potential is discretized on the *main* control volume with second-order central differences

$$\frac{\phi_{i+1,j,k}+\phi_{i-1,j,k}}{\delta x^2}+\frac{\phi_{i,j+1,k}+\phi_{i,j-1,k}}{\delta y^2}+\frac{\phi_{i,j,k+1}+\phi_{i,j,k-1}}{\delta z^2}-2\phi_{i,j,k}\left(\frac{1}{\delta x^2}+\frac{1}{\delta y^2}+\frac{1}{\delta z^2}\right)=D, \qquad (20)$$

where $\phi_{i+1,j,k}$, $\phi_{i-1,j,k}$, $\phi_{i,j+1,k}$, $\phi_{i,j-1,k}$, $\phi_{i,j,k+1}$, $\phi_{i,j,k-1}$, and $\phi_{i,j,k}$ represent the pressure potential function in the *right*, *left*, *top*, *bottom*, *far*, *in*, and *i,j,k* macro cells, respectively; and $D=\frac{\partial\tilde{u}}{\partial x}+\frac{\partial\tilde{v}}{\partial y}+\frac{\partial\tilde{w}}{\partial z}$ is the incompressibility deviation function.

A macro cell in the vicinity of the free surface is presented in Fig. 8. The distance in the x-direction between the center of macro cell (i,j,k), where the pressure potential function is computed, and the free

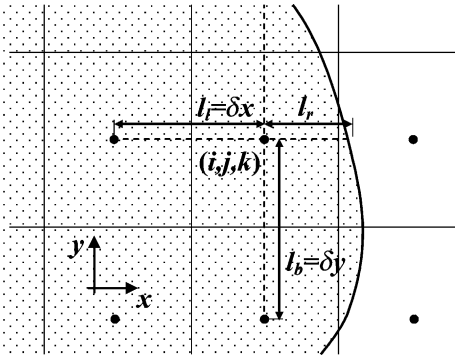


Fig. 8. Points used in calculating the pressure potential function.

surface, where a correct boundary condition must be applied, is not equal with δx. For this reason, Eq. (20) is rewritten to account for unequal leg lengths

$$\frac{\phi_{i+1,j,k}}{l_r\delta x}+\frac{\phi_{i,j+1,k}}{l_t\delta y}+\frac{\phi_{i,j,k+1}}{l_o\delta z}+\frac{\phi_{i-1,j,k}}{l_l\delta x}+\frac{\phi_{i,j-1,k}}{l_b\delta y}+\frac{\phi_{i,j,k-1}}{l_i\delta z}$$
$$-\phi_{i,j,k}\left(\frac{1}{l_r\delta x}+\frac{1}{l_l\delta x}+\frac{1}{l_t\delta y}+\frac{1}{l_b\delta y}+\frac{1}{l_o\delta z}+\frac{1}{l_i\delta z}\right)=D, \tag{21}$$

where l_r, l_l, l_t, l_b, l_o and l_i represent the leg lengths for macro cell (*i,j,k*). The procedure of transforming Eq. (20) into Eq. (21) is presented in detail in Johnson et al. [18]. In the ELMMC method, two types of pressure boundary conditions can be imposed by the use of initial settings for an outlet region, namely, *open* and *continuing domain*. The former boundary condition simulates an empty zone just outside the computational domain. From a mathematical point of view, this boundary condition is obtained by setting the pressure potential function equal to zero on the interface between the *empty* and the *outlet* macro cells.

The *continuing domain* pressure boundary condition is equivalent with setting the pressure potential function gradient equal to zero, and is used to simulate an infinite channel or an open sea. Also, the *continuing domain* condition is used to set the pressure boundary condition for the *inlet* cells.

The zero gradient condition is also used as the pressure boundary condition for *wall* cells. For thin walls and for wall corners, a multi-valued external pressure potential field is used. A very thin solid wall surrounded by *empty* macro cells is presented in Fig. 9. In order to correctly apply the zero pressure gradient

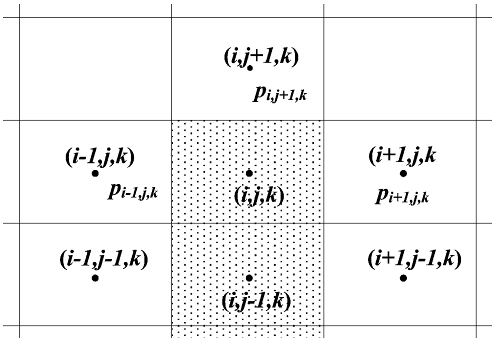


Fig. 9. Velocity boundary conditions.

condition, the potential pressure value inside cell (*i,j,k*) must equal to either $p_{i-1,j,k}$, $p_{i,j+1,k}$, or $p_{i+1,j,k}$, depending on which of these cells the boundary condition is being imposed. In the ELMMC method, the pressure boundary conditions are imposed by modifying the leg length for each empty cell and not by setting pressure values outside the computational domain.

The boundary condition on the free surface is set to be equal with the free surface tension. Estimation of the free surface tension is based on the approximation that the local shape of the surface can be considered as part of an ellipsoid. For the macro cells that intersect the free surface, the free surface tension that acts on the control volume (*i,j,k*) is computed from

$$f_x = \alpha\left(\frac{1}{R_{xoy}} + \frac{1}{R_{xoz}}\right), \tag{22}$$

where α is the surface tension coefficient and R_{xoy} and R_{xoz} are approximations for the radii of curvature of the free surface [4,11,21].

Computing R_{xoy} and R_{xoz} involves two steps. First, a search for all markers contained in macro cell *i,j,k* is performed to determine the closest marker from the *ox*-axis. This would be marker *A* in the case shown in Fig. 10. By projecting this marker on the *ox*-axis, a new point, *A**, is obtained. In the second step, all markers contained in macro cell *i, j, k* are searched to determine the closest markers from planes *xoy* and *xoz*. Then, the projections of these closest markers on planes *xoy* and *xoz* are calculated, retaining only the farthest two projected markers for each plane. Based on these points, the radii of curvature R_{xoy} and R_{xoz} are computed in each plane considering the intersection of the free surface with each plane to be a circular arc. Once the boundary conditions have been appropriately set, the pressure Poisson equation is solved by the use of a highly efficient and robust preconditioned conjugate gradient method.

2.3.6. Computation of the final velocity field

The final velocity field is obtained by correcting the tentative velocity field with the potential pressure field. In this step, the velocity type is also set based on the macro cells flag. Based on the neighbor cell flag, the six main types of velocities are illustrated in Fig. 11.

2.3.6.1. Internal velocities. Velocities $\bar{u}_{i-1,j-1,k}$ and $\bar{u}_{i-1,j,k}$ are immersed in the fluid, and thus their flag is named *internal*. However, there is a difference in the pressure correction formula used in these two cases. The velocity $\bar{u}_{i-1,j,k}$ has a *surface* cell on its right, where the potential pressure function has not been

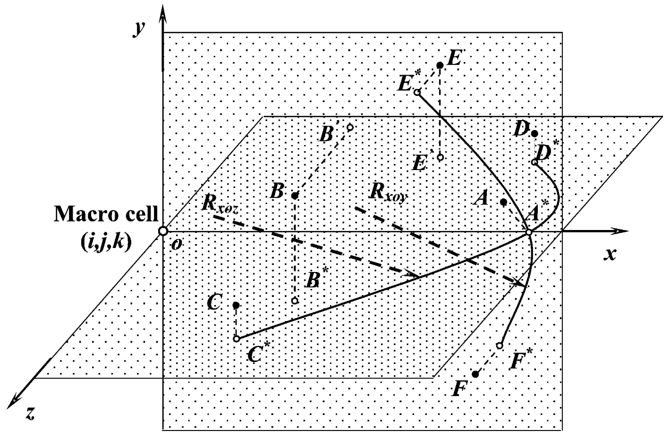


Fig. 10. Projection of markers in the computation of free surface tension.

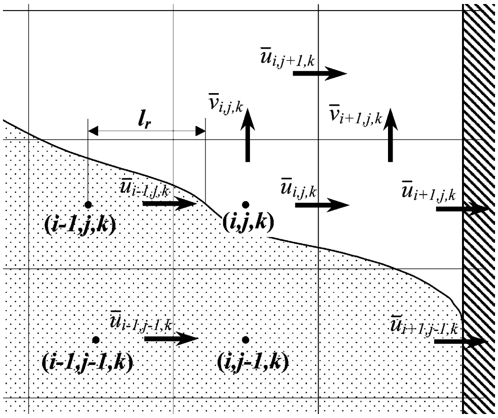


Fig. 11. Final velocity types.

calculated. On the other hand, velocity $\bar{u}_{i-1,j-1,k}$ is located between two *full* cells. The final velocity $\bar{u}_{i-1,j-1,k}$ is computed according to

$$\bar{u}_{i-1,j-1,k} = \tilde{u}_{i-1,j-1,k} - \frac{\phi_{i,j-1,k} - \phi_{i-1,j-1,k}}{\delta x} \tag{23}$$

while velocity $\bar{u}_{i-1,j,k}$ is calculated according to

$$\bar{u}_{i-1,j,k} = \tilde{u}_{i-1,j,k} - \frac{\delta t f_x - \phi_{i-1,j,k}}{l_r} \tag{24}$$

where f_x is the free surface tension and l_r is the right leg length for cell $(i-1,j,k)$.

2.3.6.2. Obstacle velocities. Velocity $\bar{u}_{i+1,j-1,k}$ is also internal, but lies between a *full* and an *obstacle* macro cell. Thus, the velocity flag is named *obstacle* and the velocity value is thus set equal to zero.

2.3.6.3. Just outside tangential velocities. All other velocities in Fig. 11 belong to the external velocity field. A correct assignment of each external velocity is critical to accurately simulating the free surface advection in the next time level. In previous methods, the external velocity field was partially assigned (SMMC) or depended on cell sweeping order (MAC). However, in the ELMMC method the entire external velocity field is calculated, which makes it the most elaborate step in the entire algorithm from a computational logic point of view. The estimation of $\bar{u}_{i,j,k}$ can be performed in 94 distinct cases based on the information from up to 26 neighbors and the position of the free surface. The flag name for the type of velocity represented by $\bar{u}_{i,j,k}$ is *just outside tangential*, and is given according to the position of the velocity relative to the free surface location.

The estimation of a *just outside tangential* velocity from a mathematical point of view is an extrapolation of the internal velocity field beyond the free surface. Like any other extrapolation procedure, this step must be performed very carefully in order not to introduce errors in the external velocity field. The problem consists in the identification of the closest appropriate information from the 26 possibilities. The following hierarchical steps accomplish the estimation of the velocity $\bar{u}_{i,j,k}$.

Step 1. Two fluid fronts are depicted in Fig. 12(a), the decision of which information to use for the *just outside tangential* velocity $\bar{u}_{i,j,k}$ is based on the flags of velocities $\bar{v}_{i,j-1,k}$ and $\bar{v}_{i+1,j-1,k}$, at least one of which must be *internal* in order to consider the velocity $\bar{u}_{i,j-1,k}$ a pertinent value. For the sake of this discussion, let $\bar{v}_{i+1,j-1,k}$ be an internal velocity. Next, a similar check is made for velocities $\bar{u}_{i,j+1,k}$, $\bar{u}_{i,j,k+1}$, and $\bar{u}_{i,j,k-1}$.

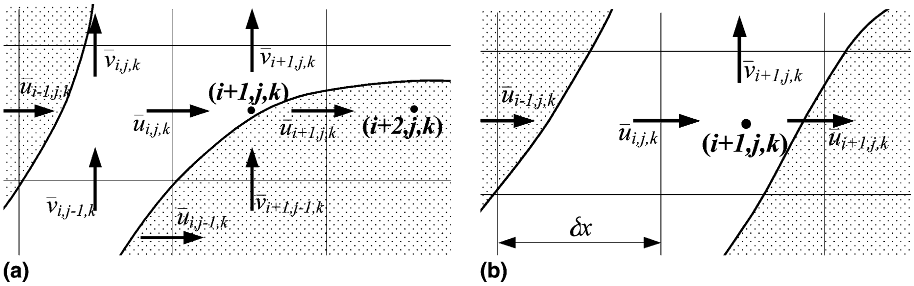


Fig. 12. First two hierarchical steps for assigning the just outside tangential velocity.

The decision is made based on the velocity flags of ($\bar{v}_{i,j,k}$ and $\bar{v}_{i+1,j,k}$), ($\bar{w}_{i,j,k}$ and $\bar{w}_{i+1,j,k}$), and ($\bar{w}_{i,j,k-1}$ and $\bar{w}_{i+1,j,k-1}$), respectively, at least one of which must be an *internal* velocity. If, based on the above algorithm, more than one direct neighbor is found, then the velocity $\bar{u}_{i,j,k}$ is computed as an averaged value between their values. If it was not possible to find in *Step 1* a value for the velocity $\bar{u}_{i,j,k}$, then *Step 2* is considered.

Step 2. For the fluid configuration shown in Fig. 12(b), the velocity $\bar{u}_{i,j,k}$ was not assigned at the end of the *Step 1*. Now only the left and the right neighbors are checked to see if they belong to the internal velocity field. The decision to use or not to use the velocities $\bar{u}_{i-1,j,k}$ and/or $\bar{u}_{i+1,j,k}$ is taken based on whether for the former the right leg length of macro cell ($i-1,j,k$) is greater than $\delta x/2$ and for the latter whether the left leg length of macro cell ($i+2,j,k$) is greater than $\delta x/2$. If at least one reliable neighbor is found, then the average value is used for the *just outside tangential* velocity $\bar{u}_{i,j,k}$. For the given configuration, the velocity $\bar{u}_{i,j,k}$ lies between different fluid fronts, and its value is important only during this time level for computing the *fluid-empty* velocities, such as $\bar{v}_{i+1,j,k}$. If it was not possible to find a value for the velocity $u_{i,j,k}$ in *Step 2*, then *Step 3* is considered.

Step 3. In this step, the search is performed among the velocities located in the same plane as velocity $\bar{u}_{i,j,k}$, but excluding the *direct* neighbors. A portion from a horizontal jet is depicted in Fig. 13. Curve *AB* represents the intersection between the free surface of the jet and plane *yoz*. The only internal velocity is $\bar{u}_{i,j-1,k+1}$, while $\bar{u}_{i,j-1,k}$, $\bar{u}_{i,j,k+1}$, and $\bar{u}_{i,j,k}$ are the *just outside tangential* velocities that must be assigned.

The only "trustworthy" information, for the computation of $\bar{u}_{i,j,k}$, is the velocity $\bar{u}_{i,j-1,k+1}$. The decision to use or not use this information is based on the flags of velocities $\bar{w}_{i,j-1,k}$, $\bar{w}_{i+1,j-1,k}$, $\bar{v}_{i,j,k+1}$ and $\bar{v}_{i+1,j,k+1}$, at least one of which must be *internal*. In addition, the other three neighbors $\bar{u}_{i,j+1,k-1}$, $\bar{u}_{i,j-1,k-1}$ and $\bar{u}_{i,j+1,k+1}$ are tested in the same way, in order to decide on the best information to extrapolate from. If, based on

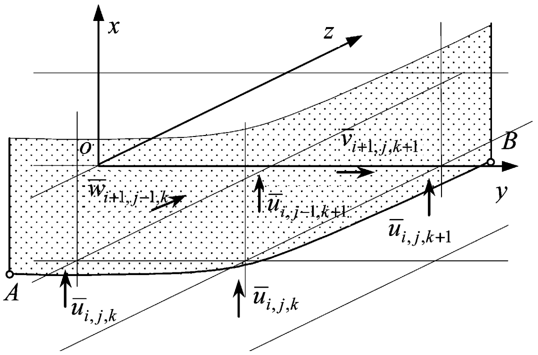


Fig. 13. Third hierarchical step for assigning the just outside tangential velocity.

the above algorithm, more than one direct neighbor is found, then the velocity is computed as an average of their values. If it was not possible to find a value for the velocity $\bar{u}_{i,j,k}$ in *Step 3*, then *Step 4* is considered.

Step 4. If the algorithm reaches this point, it means that the distance between the free surface and the velocity location is greater than $\delta x/2$, $\delta y/2$ or $\delta z/2$. The future extrapolation for $\bar{u}_{i,j,k}$ repeats the above three steps, but with less mandatory conditions. In this step, the search is for reliable information among the *direct* neighbors. The decision to use or not a velocity is based only on its velocity flag, which must be an *internal*. If more than one reliable neighbor is found, then the velocity is computed as an average of their values. If it was not possible to find a value for velocity $\bar{u}_{i,j,k}$ in *Step 4*, then *Step 5* is considered.

Step 5. In this step, the test is for neighboring velocities located in the same plane with velocity $\bar{u}_{i,j,k}$, but excluding direct neighbors and the same velocities as in *Step 3* are checked. The decision to use one of velocities $\bar{u}_{i,j+1,k+1}$, $\bar{u}_{i,j+1,k-1}$, $\bar{u}_{i,j-1,k-1}$, and/or $\bar{u}_{i,j+1,k+1}$ is based on their velocity flags, which must be *internal*. If more than one reliable neighbor is found, then the velocity $\bar{u}_{i,j,k}$ is computed as an average. If it was not possible to find a value for velocity $\bar{u}_{i,j,k}$ in *Step 5*, then *Step 6* is considered.

Step 6. In this final step, the search is among the last 16 neighbors. The decision to use one of them is based on their velocity flag, which must be *internal*. If more than one reliable neighbor is found, then the velocity $\bar{u}_{i,j,k}$ is computed as an average. If it was not possible to find a value for velocity $\bar{u}_{i,j,k}$ in *Step 6*, then its velocity flag is switched from *just outside tangential* to *outside the fluid*. Obviously, in the next computational cycle, if a velocity value is necessary for the $\bar{u}_{i,j,k}$ location, then a value is assigned temporarily from the external velocity field.

2.3.6.4. Fluid-empty velocities. Fluid-empty velocities lie between a *surface* and an *empty* macro cell. Of the velocities shown in Fig. 11, $\bar{u}_{i+1,j,k}$, $\bar{v}_{i,j,k}$, and $\bar{v}_{i+1,j,k}$, are of the *fluid-empty* type. Their values are computed after all the *just outside tangential* velocities have been set. The physical concept behind the computation of *fluid-empty* velocities is to satisfy the continuity equation (Eq. (2)), in *surface* cells, with exceptions allowed for few special cases like for convergent fluid fronts. This stage is not a straightforward process since there are 63 distinct cases in which the *fluid-empty* velocities are computed. It is worth mentioning that a major difference exists in the computation of the *fluid-empty* velocities as compared to the *just outside tangential* velocities. All *fluid-empty* velocities belonging to the same macro cell are computed simultaneously, whereas for the *just outside tangential* velocities, each case is judged independently.

Based on the total number of *fluid-empty* velocities for a given macro cell, 63 different possibilities exist. The *fluid-empty* velocity $\bar{u}_{i,j,k}$, presented in Fig. 14(a), is the only one that belongs to the external velocity field; all other velocities associated with macro cell (i,j,k) are internal velocities. In all similar cases velocity $\bar{u}_{i,j,k}$ is computed by the use of the continuity equation

$$\bar{u}_{i,j,k} = \bar{u}_{i-1,j,k} - \left(\bar{v}_{i,j,k} - \bar{v}_{i,j-1,k}\right)\frac{\delta x}{\delta y} - \left(\bar{w}_{i,j,k} - \bar{w}_{i,j,k-1}\right)\frac{\delta x}{\delta z}. \tag{25}$$

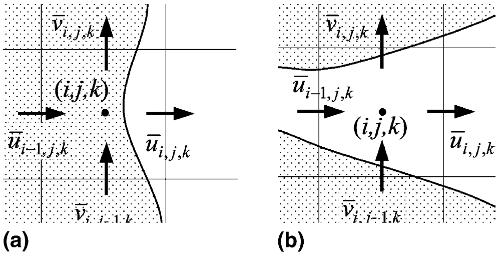


Fig. 14. Cell case with one velocity of type fluid-empty.

An example on an exception is presented in Fig. 14(b), where velocity $\bar{u}_{i,j,k}$ is located between two converging fluid fronts. If the same relation is used to estimate $\bar{u}_{i,j,k}$, a build up process would occur and in just a few time levels the absolute value of $\bar{u}_{i,j,k}$ will rise dramatically. For this reason, in such cases, the value of the velocity $\bar{u}_{i,j,k}$ is assigned equal to $\bar{u}_{i-1,j,k}$.

Another representative example is presented in Fig. 15, where two velocities that belong to the same macro cell possess the *empty-surface* flag. The first case presented in Fig. 15(a) represents a fluid front that has just entered in the macro cell, with all velocities belonging to the external velocity field. The best information available for velocities $\bar{u}_{i,j,k}$ and $\bar{v}_{i,j,k}$ should be $\bar{u}_{i-1,j,k}$ and $\bar{v}_{i,j-1,k}$, respectively. Nevertheless, both of the latter velocities are extrapolated values, and the information is thus not "trustworthy". In this case, it is no longer essential to satisfy the continuity equation for this macro cell. Instead, the *empty-surface* velocities, $\bar{u}_{i,j,k}$ and $\bar{v}_{i,j,k}$, are assigned equal to $\bar{u}_{i-1,j,k}$ and $\bar{v}_{i,j-1,k}$, respectively.

In the next time level, the fluid front advances and its position could be as depicted in Fig. 15(b). The velocity $\bar{v}_{i,j-1,k}$ is now residing within the internal velocity field. First, the reliable information is extrapolated to $\bar{v}_{i,j,k}$ and then the continuity condition is used to compute $\bar{u}_{i,j,k}$.

The next case depicted in Fig. 15(c) is when both velocities $\bar{u}_{i,j,k}$ and $\bar{v}_{i,j,k}$ are within the internal velocity field. In this case, the "trustworthy" information is first extrapolated. Then, the deviation from continuity is computed for this control volume. The velocities $\bar{u}_{i,j,k}$ and $\bar{v}_{i,j,k}$ are corrected so that the continuity equation is satisfied in this macro cell

$$\begin{aligned} D &= \frac{\bar{w}_{i,j,k} - \bar{w}_{i,j,k-1}}{2\delta z}, \\ \bar{u}_{i,j,k} &= \bar{u}_{i,j,k} - D\delta x, \\ \bar{v}_{i,j,k} &= \bar{v}_{i,j,k} - D\delta y. \end{aligned} \tag{26}$$

The last case depicted in Fig. 15(d) is that of two converging fluid fronts, where the *fluid-empty* velocities are located on opposite faces of the same macro cell. This case is also resolved based on the use of the continuity equation. First, a deviation from continuity is computed, and then this value is used to correct velocities $\bar{u}_{i,j,k}$ and $\bar{v}_{i,j,k}$

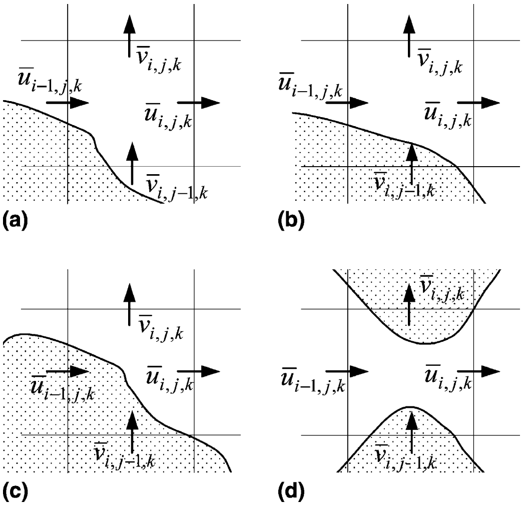


Fig. 15. Case 2: The fluid-empty velocities.

$$
\begin{aligned}
&D = \frac{1}{2}\left(\frac{\bar{u}_{i,j,k} - \bar{u}_{i-1,j,k}}{\delta x} + \frac{\bar{v}_{i,j,k} - \bar{v}_{i,j-1,k}}{\delta y} + \frac{\bar{w}_{i,j,k} - \bar{w}_{i,j,k-1}}{\delta z}\right),\\
&\bar{u}_{i,j,k} = \bar{u}_{i,j,k} - D\delta x, \;\; \bar{u}_{i-1,j,k} = \bar{u}_{i-1,j,k} + D\delta x.
\end{aligned}
\tag{27}
$$

The other 61 cases for *fluid-empty* velocities, considered in the ELMMC method, are solved with similar approaches. The above algorithm is more complicated when three, four or five *fluid-empty* velocities belong to the same macro cell.

2.3.6.5. Outside the fluid velocities. The last type of velocity depicted in Fig. 11 is represented by $\bar{u}_{i,j+1,k}$, which lies between *empty* macro cells. The flag name for this velocity is *outside the fluid*, and its value is set equal to zero. In the next time level, if velocity information is needed at this location, then a temporary value is extrapolated from the internal velocity field.

2.3.7. Computation of eddy viscosity

This is the final step in the computational cycle, and is done only if the k–ε turbulence model is activated as part of a simulation. The kinetic energy equation and the dissipation of the kinetic energy equation are discretized on the *main* control volume. The convective terms are approximated by the use of a control volume approach while the other terms are approximated by a modified finite difference approach

$$
\frac{\partial k}{\partial t} + \bar{u}_j \frac{\partial k}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\frac{\nu_t}{\sigma_k}\frac{\partial k}{\partial x_j}\right) + \nu_t \frac{\partial \bar{u}_i}{\partial x_j}\left(\frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_j}{\partial x_i}\right) - \varepsilon,
\tag{28}
$$

$$
\frac{\partial \varepsilon}{\partial t} + \bar{u}_j \frac{\partial \varepsilon}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\frac{\nu_t}{\sigma_\varepsilon}\frac{\partial \varepsilon}{\partial x_j}\right) + C_{1\varepsilon}\frac{\varepsilon}{k}\nu_t \frac{\partial \bar{u}_i}{\partial x_j}\left(\frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_j}{\partial x_i}\right) - C_{2\varepsilon}\frac{\varepsilon^2}{k}.
\tag{29}
$$

The discrete approximations of Eqs. (28) and (29) must minimize numerical diffusion, and reduce the stiffness inherent in the production and dissipation terms. In addition, because the final velocity field is already available, when the eddy viscosity is computed, a semi-implicit scheme (see [22]) is used for time discretization, which has the advantage of increasing the stability of the turbulent kinetic energy solution. The final discrete forms of the k–ε turbulence model equations appear as

$$
\begin{aligned}
&\varepsilon^{n+1}_{i,j,k} \approx \frac{1}{\frac{1}{\delta t} + C_{2\varepsilon}\left(\frac{\varepsilon^n_{i,j,k}}{k^n_{i,j,k}}\right)}\left[\frac{\varepsilon^n_{i,j,k}}{\delta t} + A_\varepsilon + C_{1\varepsilon}\left(\frac{\varepsilon^n_{i,j,k}}{k^n_{i,j,k}}\right)\Re^{n+1}_{i,j,k}\right],\\
&k^{n+1}_{i,j,k} \approx k^n_{i,j,k} + \delta t\left[A_k + 0.5\left(\Re^{n+1}_{i,j,k} + \Re^n_{i,j,k} - \varepsilon^{n+1}_{i,j,k} - \varepsilon^n_{i,j,k}\right)\right],
\end{aligned}
\tag{30}
$$

where A_k is a generic name for all the terms in Eq. (28) that contain the turbulent kinetic energy, k; A_ε is a generic name for all the terms in Eq. (29) that contain the dissipation rate of the turbulent kinetic energy, ε; and $\Re$ is a generic name for all the terms in Eqs. (28) and (29) that do not contain the turbulent kinetic energy or the dissipation rate of the turbulent kinetic energy.

In a macro cell adjacent to a solid boundary, the local state of the turbulence is expressed in terms of the friction velocity and computed implicitly from the logarithmic law of the wall for a smooth boundary. Based on the resulting friction velocity, the boundary conditions for k and ε are computed by the use of the van-Driest law, which accounts for the direct viscous dissipation when the turbulence intensity is low. For an outflow boundary, a zero gradient continuation condition is applied for k, ε, and ν_t. For a cell in the vicinity of the free surface, the k–ε model is applied only when the cell's central micro cell carries a *full* computational flag. In order to derive simple free surface boundary conditions for k and ε, assumptions are made that all transport processes that occur within the body of water are driven by internal shear, that the influence of the external fluid ("air") is neglected, and that the turbulence does not diffuse across the free surface interface. The practical implication of this approach is that the gradients of the turbulent quantities die out along the local normal to the free surface.

It is important to point out that the treatment of free surface turbulence in the types of complex flows for which the ELMMC method was envisioned is a non-trivial undertaking, and as such is beyond the scope of this work. The intent of including the k–ε model and its implementation in this work was only to demonstrate the capability of the ELMMC framework to accommodate such physics, if and when needed.

3. Validation of the ELMMC method

The first example involves the interaction of a single, large wave with a tall structure. The animated simulation results are shown in Fig. 16 in the form of a sequence of still images of the free surface, beginning with the initial condition at time equal zero. The experimental results for this problem were produced at the University of Washington, Seattle (data provided by Profs. Catherine Petroff and Harry Yeh). The tank under consideration is 1.6 m long, 0.61 m wide, and 0.75 m high. The volume of water initially contained behind a gate is 0.4 m × 0.61 m × 0.3 m. The structure, which is 0.12 m × 0.12 m × 0.75 m, is placed 0.5 m downstream of the gate and 0.24 m from the near sidewall of the tank. In the physical experiment, since it is impossible to completely drain the tank downstream of the gate, a layer of water (approximately 0.01 m deep) is present on the bottom of the tank. The domain is discretized with macro cells of dimensions 0.02 m × 0.01 m × 0.01 m, resulting in a computational domain composed of 80 × 61 × 75 cells. Surface cells are subdivided into 27 micro cells, three in each spatial direction.

The initial condition for this problem is presented in the first frame of Fig. 16. Once the gate is removed at time equal to zero, gravity sets the water in motion toward the structure. The center part of the water front impacts the structure at $t \approx 0.4$ s and rides up its upstream face. Meanwhile, the sides of the wave front wrap around the structure, rejoin in its wake, and proceed toward the downstream wall of the tank.

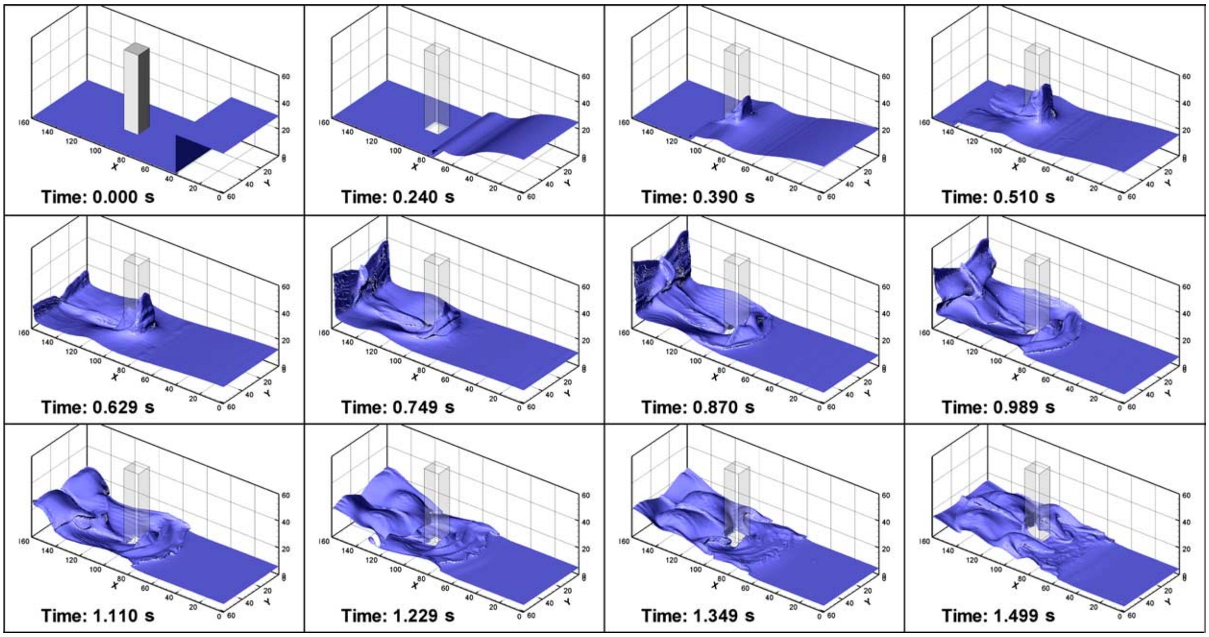


Fig. 16. Simulation of wave impact with a tall structure.

A “focusing” effect is observed in the wake of the structure, with recirculation behavior normally indicative of strong scouring. The mass of water rides up the downstream wall of the tank, and then comes crashing onto the bottom of the tank. The strengths of the computational technique in dealing with impact, wave breaking, jets, and converging fluid fronts are well highlighted by this complex flow problem. While the simulation appears to be visually realistic, it is important to assess the accuracy of the numerical results.

Collected measurements include the time history of the net force on the structure, and the time history of the fluid velocity at different locations. Forces were measured with a load cell mounted inside the tall structure at its bottom, and velocities were measured with a laser-Doppler velocimetery (LDV) system. The velocity measurements were performed at three locations. The first location is at 0.146 m upstream of the center of the structure and 0.026 m off the floor of the tank, the second location is at 0.088 m downstream of the center of the structure and 0.05 m off the floor of the tank, and the last location is at 0.268 m downstream of the center of the structure and 0.035 m off the floor of the tank. The validation results for the horizontal and vertical velocities in the main flow direction are presented in Fig. 17. Data from four independent experiments are represented in the graphs as circular symbols. For each of the three pairs of graphs, the time origin is set to coincide with the moment at which the water first reaches the measurement location considered. Specifically, the time delays between the raising of the gate and the instant when the water first reaches the three measurement locations are 0.238, 0.5167, and 0.4512 s, respectively.

The gaps in the experimental data (e.g., 0.6 < t < 0.85 s in the top left graph of Fig. 17) are due to the presence of bubbles in the water, which scatter the laser light and degrade the signal-to-noise ratio of the

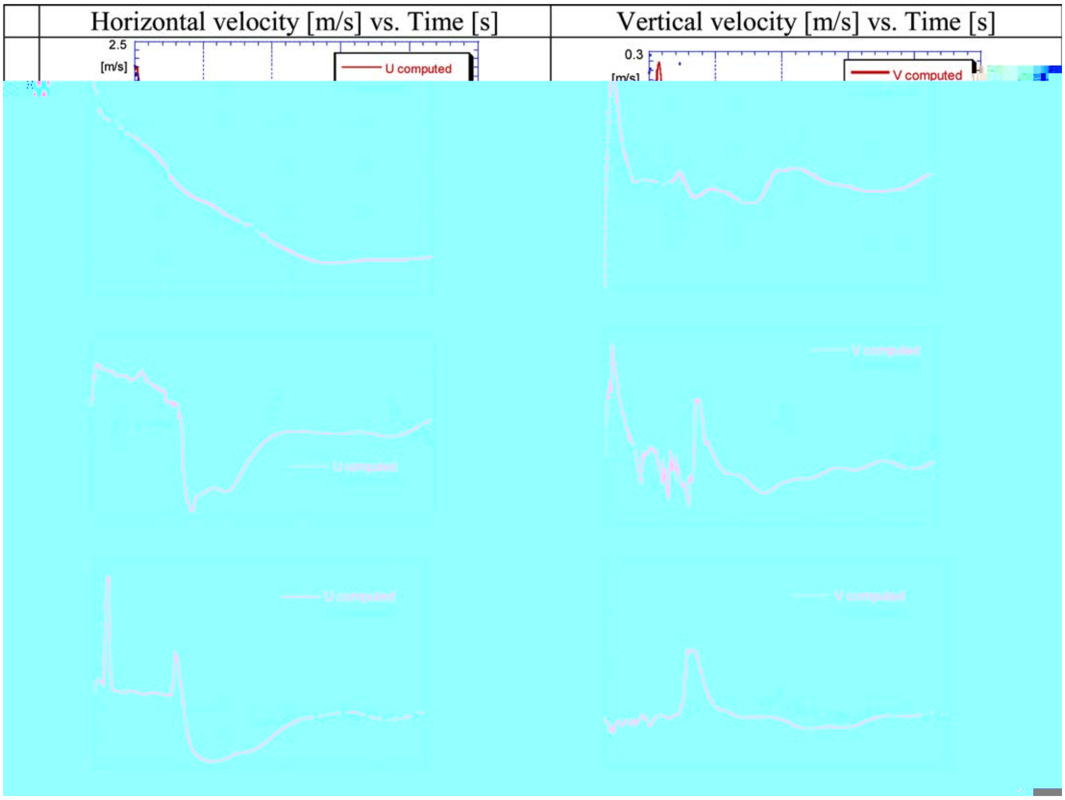


Fig. 17. Validation with experimental data for velocity field.

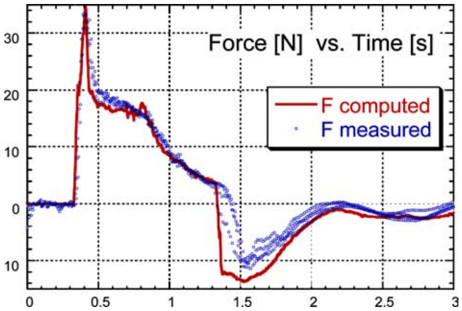


Fig. 18. Validation with experimental data for the net force.

LDV measurement system. This is a major limitation of the available experimental data. Hence, the comparison is restricted only to the horizontal velocity in location one and in location three. New experiments are being performed in which LDV and PIV measurements are made through the bottom of the tank in order to minimize the effect of laser scattering from the air bubbles on the free surface. These results and comparisons with the numerical results for several structures shapes will be reported on elsewhere once the new investigations have been completed. The numerical results obtained for the vertical velocity at the second location show a chattering behavior at around $t = 0.5$ s. This is due to the external velocity field where the information is extrapolated based on different logic from one step to the next. Once the point falls inside the fluid, the oscillations cease and normal variations in the velocity field are observed.

The validation results for the net force on the structure are presented in Fig. 18. The numerical results are drawn with a solid line and the experimental data, obtained from four independent experiments, are plotted as circular symbols. The net numerical force is calculated by summing the forces that act on each boundary cell along the front and back faces of the structure. The force in each boundary cell is calculated as the product of the pressure at the center of that cell and the cell face area. The force due to the shear stresses on the lateral faces of the structure is negligible compared to the forces on the front and back faces. [1] The numerical results for the initial impact, which took place at around 0.38 s, coincide nicely with the experimental data. However, an undershooting occurs at the second impact, which takes place at around 1.35 s. The explanation for the latter discrepancy is the existence in the experiments of a large quantity of air bubbles entrained in the water during the wave breaking downstream of the structure. The bubbles decrease the local fluid density, and in turn the impact force. The air bubble effect is unaccounted for in the numerical simulation.

Overall, the comparisons between the experimental and numerical results for the net force and the horizontal velocity provide strong confidence in the validity and consistency of the ELMMC method.

4. Accuracy and convergence

The accuracy of the numerical results is assessed by performing spatial and temporal convergence studies on the method. To demonstrate the stability of the ELMMC method, a new problem is chosen, representing a spillway geometry as depicted in Fig. 19, where all dimensions are in meters. The computational domain is divided into two chambers by a vertical ‘‘U’’-shaped wall; the left side of the domain is the inlet chamber and the right side is the outlet region. Initially, the computational domain is empty. The fluid

[1] Even if calculated with the maximum values of velocity and surface area, the total force amounts to 0.03 N

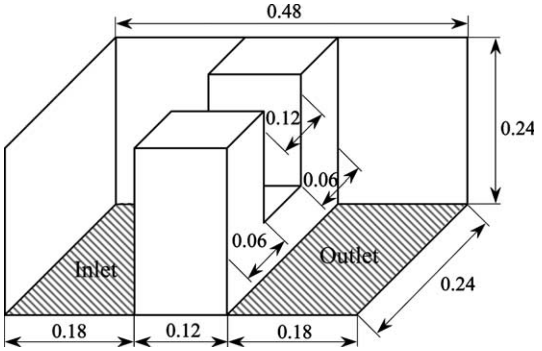


Fig. 19. Geometry of the spillway problem.

enters through the entire bottom of the inlet chamber with a velocity of 0.25 m/s. As the water level in the inlet chamber rises, the water eventually flows into the outlet chamber through the spillway channel in the middle of the vertical wall. Stationary flow is achieved when a balance is reached between the gravitational and inertial forces along the channel. This spillway problem represents a useful verification problem of some computational complexity. Particularly challenging to the ELMMC method should be the handling of the jet as it goes through the outlet chamber since marker advection is based partially on the external velocity field. All numerical computations discussed next were performed without activating the turbulence model.

To demonstrate spatial convergence, seven incrementally finer grid resolutions were considered with computational domains of 32 × 16 × 16, 40 × 20 × 20, 48 × 24 × 24, 64 × 32 × 32, 72 × 36 × 36, 80 × 40 × 40, and 96 × 48 × 48 grid cells. The number of grid cells increases from one simulation to the next

Table 1
Grid convergence results and associated computational costs for the spillway problem

Representative quantities		Cases						
		32 × 16 × 16	40 × 20 × 20	48 × 24 × 24	64 × 32 × 32	72 × 36 × 36	80 × 40 × 40	96 × 48 × 48
Grid cells		8192	16,000	27,648	65,536	93,312	128,000	221,184
	%	–	95.3	72.8	137.0	42.4	37.2	72.8
Froude number		0.6192	0.6544	0.6641	0.6786	0.6880	0.6835	0.6825
	%	–	5.7	1.5	2.2	1.4	0.65	0.15
Average flow velocity	(m/s)	0.7982	0.8221	0.8448	0.8539	0.8674	0.8632	0.8627
	%	–	2.3	2.8	1.1	1.6	0.48	0.06
Flow area	(m^2)	0.02032	0.01930	0.01979	0.01937	0.01944	0.01950	0.01954
	%	–	5.0	2.5	2.1	0.38	0.33	0.19
Water level	(m)	0.20933	0.20312	0.20785	0.20557	0.20767	0.20739	0.20698
	%	–	2.3	2.3	1.1	1.0	0.13	0.20
Shooting point	(m)	0.44015	0.44914	0.45969	0.45271	0.45471	0.45773	0.45804
	%	–	2.0	2.4	1.5	0.33	0.77	0.07
Number of time steps		320	365	463	570	667	825	932
	%	–	14.0	26.8	23.1	27.2	23.7	13.0
CPU time	(s)	81.2	155.7	420.7	1011.6	1973.1	3434.7	8203.6
	%	–	91.8	170.3	140.3	95.0	74.1	138.8

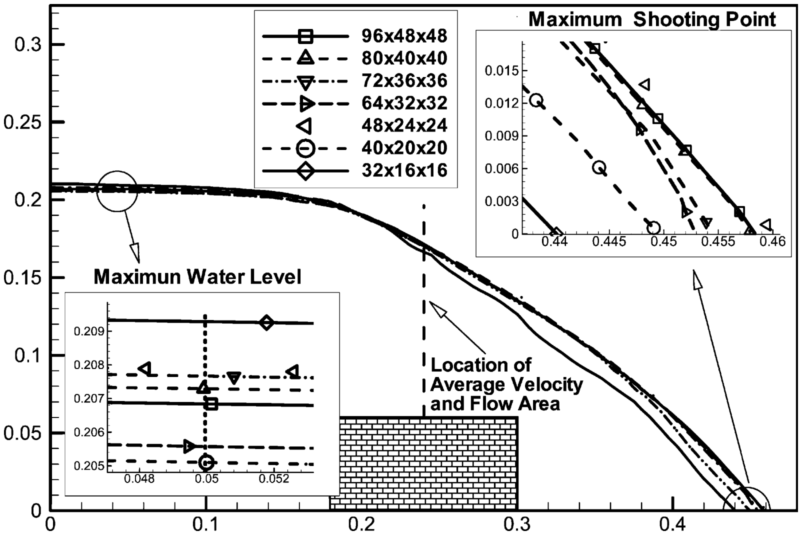


Fig. 20. Locations and values of the representative quantities.

by at least 37%, with the maximum relative increase of 137% taking place for grid case 64 × 32 × 32. The overall increase in the number of cells is 2600%. The Froude number, average flow velocity and flow area (computed along a vertical plane normal to the flow and cutting through the middle of the spillway), maximum water level in the inlet chamber, and maximum shooting point of the jet in the outlet chamber are presented in Table 1 for the seven grid resolutions considered. The measurement locations and the final position of the free surface are depicted schematically in Fig. 20. For each representative quantity, Table 1 lists both the computed value and the relative change in that quantity from the previous case. To get a perspective on the computational efficiency of the method, it is useful to also examine the rate at which the number of time steps required and computational time expended increase with grid resolution. These data are listed at the bottom of Table 1, and will be discussed after the physical results are addressed.

For grid case 40 × 20 × 20, the relative increase in the total number of grid points from the base case is 95.3%. The maximum variation in the computed values (5.68%) is seen to occur for the *Froude number*. For the next grid case (48 × 24 × 24), which represents an increase of 72.8% in the number of cells, the maximum relative change (2.8%) occurs in the *average flow velocity*. As mentioned above, grid case 64 × 32 × 32 introduces the maximum relative increase in the number of grid cells (137.04%). The maximum relative change (2.2%) again occurs for the *Froude number*. The next case increases the total number of cells by 42.4% (to 93,312) and yields a maximum relative change of 1.6% in the *average flow velocity*. The next case (80 × 40 × 40) yields a further reduction in the maximum relative change; namely, 0.77% for the *shooting point*. The last case considered has 221,184 computational cells, which represents a relative increase of 72.8% from the penultimate case. For this final case, the maximum change in the computed values is 0.2%, and is obtained for the *water level* in the inlet chamber.

Collectively, the results indicate that grid convergence is achieved for all five representative physical quantities that were tracked over the seven grid resolutions considered. Further examination of the trends reveals that while the final levels achieved are quite small, the rate of change for a given physical quantity does not always decrease monotonically. A fundamental reason for this behavior rests in the fact that in Eulerian–Lagrangian methods, such as the ELMMC, the spatial and temporal resolutions are coupled. In other words, it is not possible to decrease the grid cell size while keeping the time step fixed. Doing so could allow a marker to travel more than the length of a micro cell, resulting in a breakdown of the flagging scheme. As discussed in Section 2.3.3.1, the time step is calculated dynamically at the conclusion of each computational cycle to satisfy several criteria of maximum allowable change. Before discussing temporal convergence, however, it is worthwhile to examine briefly the computational costs associated with changes in the spatial resolution (bottom two rows of Table 1).

For consistency in the comparisons, the tabulated values were recorded over a fixed duration of one second of physical time, starting at $t = 0.2$ s. The latter beginning physical time was chosen because it approximately coincides with the phase in the simulation when the water jet makes contact with the bottom floor of the outlet chamber. Selecting such an initial time avoids accounting for different initialization costs and ensures that the number of iterations in the pressure solver remains nearly constant. Of course, the computational costs associated with each pressure iteration are higher for finer grids because the matrices are larger. Therefore, the tabulated computational costs should be viewed as comparative and not representative of the total costs of a simulation. Overall, a 27-fold increase in the size of the grid results in only a threefold increase in the number of pressure solver iterations, but a 100-fold increase in the computational processing time. The relatively small increase in the number of iterations is indicative of the advantageous convergence behavior of the Incomplete Cholesky PCG method being used to solve the pressure Poisson equation. It is also worth pointing out the overall efficiency of the ELMMC method, whereby the computational (wall clock) costs for a second of physical simulation time are approximately only two hours even on a 660 MHz Alpha-processor workstation.

Next, a time convergence study is presented for a chosen grid resolution case of $80 \times 40 \times 40$ (to keep CPU time manageable). At each time level, the time increment calculated dynamically by the ELMMC method was divided by two, three, or four. Hence, the flow was simulated with twice, three and four time the number of time steps that would have normally been required by the algorithm methodology. The results are summarized in Table 2. When the time step is divided by two, the maximum relative change in the computed values (1.81%) is obtained for the Froude number. The Froude number again exhibits the maximum relative change (1.59%) when the time step is divided by three. For the last case, when the time step is divided by four, the maximum relative change in the computed values (0.52%) occurs in the resulting value for the water level in the inlet chamber. Collectively, the results indicate that temporal convergence is achieved for all five representative physical quantities that were tracked over the four time step resolutions considered.

Finally, based on the above results of the spatial and temporal convergence studies, it is possible to conclude that the EMMC method exhibits good spatial and temporal convergence characteristics.

To further highlight the effectiveness and accuracy of the ELMMC method, a surface tension problem in zero gravity is briefly considered. The method is also capable of solving inviscid problems without modifications. However, these demonstrations would extend the scope of this article beyond a reasonable size. The intent here is simply to highlight the ability of the method to handle complex boundary conditions since problems involving surface tension can present particular convergence issues. The chosen test problem involves a droplet of water, whose volume is 5.0 pl, that has been initially pulled along one of its axes until

Table 2
Temporal convergence results for the spillway problem

Representative quantities	Δt	$\Delta t/2$		$\Delta t/3$		$\Delta t/4$	
		Value	% Change	Value	% Change	Value	% Change
Froude number	0.68253	0.67040	1.81	0.68125	1.59	0.67785	0.50
Spillway velocity	0.86277	0.85344	1.09	0.86173	0.96	0.85868	0.36
Flow area	0.01954	0.01982	1.40	0.01957	1.29	0.01962	0.29
Water level	0.20698	0.21077	1.80	0.20868	1.00	0.20978	0.52
Shooting point	0.45804	0.46321	1.12	0.46657	0.72	0.46428	0.49

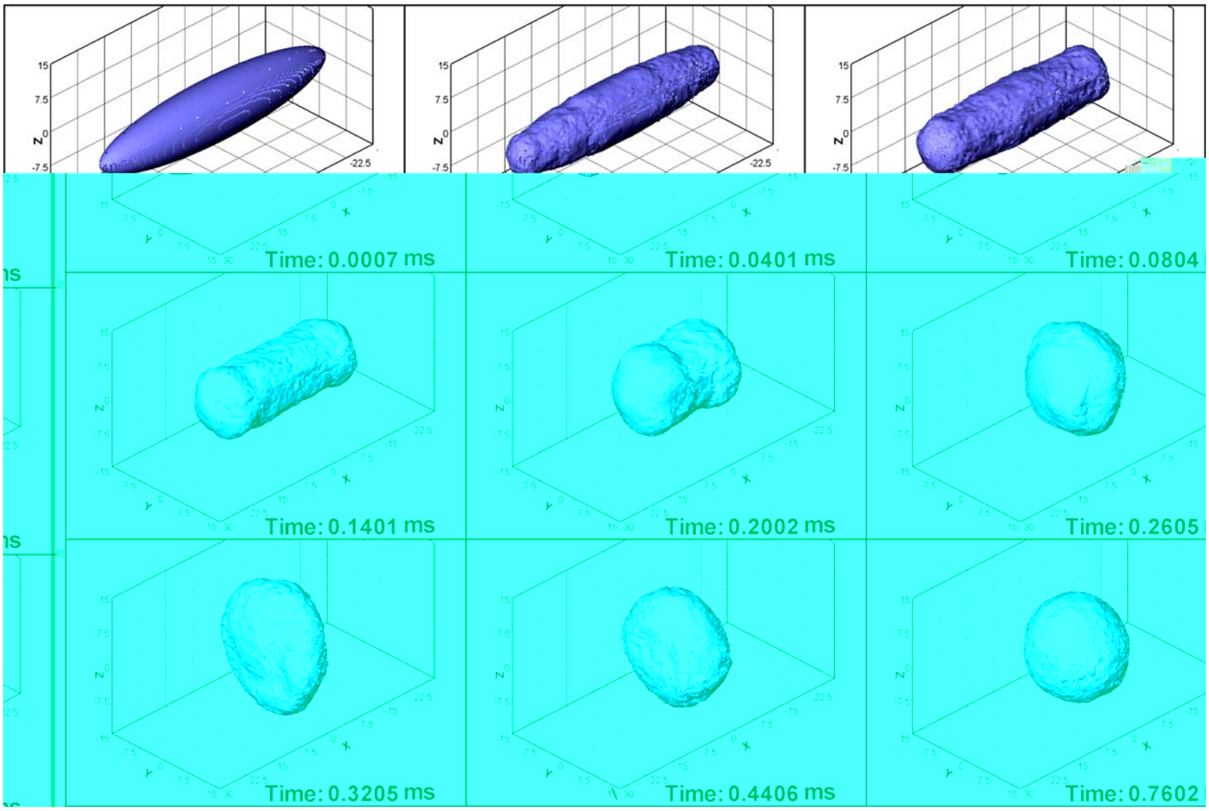


Fig. 21. Simulation of severe deformations for a water droplet in zero gravity.

its length is five times its mid-length height (i.e., diameter). This initial "cigar-shaped" condition is shown in the top leftmost frame of Fig. 21 and has the initial dimensions of 62 μm in the *x*-direction and 12.4 μm in each of the *y*- and *z*-directions. The final diameter of the droplet, for a perfect spherical shape, is 21.2 μm. After 1.4 ms, the droplet approaches its final spherical shape, with dimensions of 21 μm in the *x*-direction and 21.3 μm in each of the other two directions.

The time evolutions of the three main diameters of the droplet are shown in Fig. 22, where the inset plot zooms in on the last 150 μs of this simulation. Fig. 21 presents the free surface of the droplet at nine stages from the initial configuration (top left) until $t = 0.7$ ms (bottom right), when a reasonably spherical shape is observable. During the remaining 0.7 ms of this simulation, it is not possible to visually distinguish the surface plots of the sphere as the three diameters continue to converge asymptotically toward their final values. The apparent ripples on the surface are due to the plotting approach, which involves post-processing the surface markers into triangular patches. The search algorithm that tracks and identifies neighboring markers is very sensitive to small movements. No search was necessary for the initial condition since the positions of the markers were computed analytically, yielding a smoother plot of the free surface (not withstanding the few "holes" created by missing panels). To assess the relative accuracy of the method in converging the droplet problem to steady state, it is preferable to use an analytical metric, such as the standard deviation of the error in the positions of the surface markers as compared with the exact spherical condition. Since for the exact solution, the distance of each marker from the center of the sphere should be

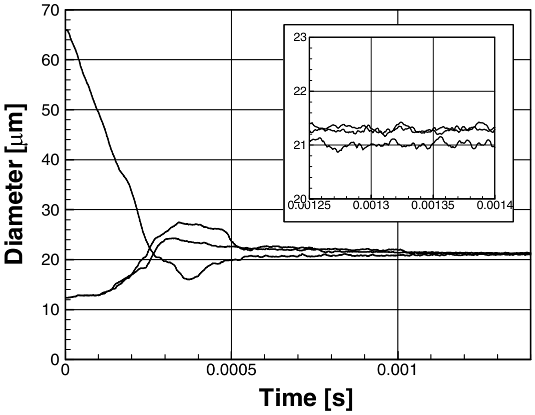


Fig. 22. Time evolution of the three droplet diameters.

21.2 μm, it is possible to calculate the actual error in the "radius" of each surface marker. The standard deviation of the errors in the marker radii was calculated to be 0.2821 μm, indicating that the method predicts a final radius for the sphere of 21.2 ± 0.3 μm.

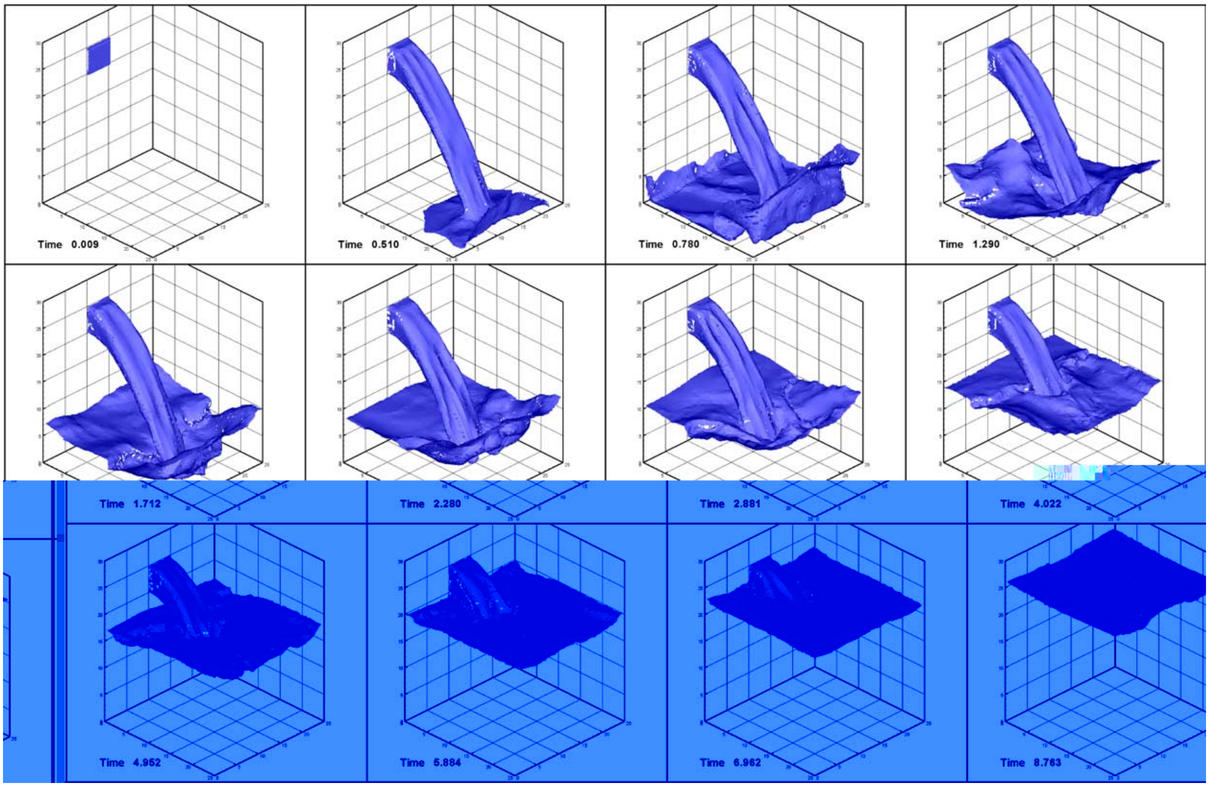


Fig. 23. Simulation of the cavity filling problem with one jet.

5. Modeling the cavity filling problem

The last examples presented next have been chosen to highlight the capabilities of the ELMMC method in simulating another important representative problem in free surface flows, namely cavity filling. Two different configurations are modeled, first that of a single jet filling a rectangular reservoir, and then second that of two jets oriented normally to each other and filling a similar reservoir.

In the first example, the computational domain consists of a rectangular reservoir with dimensions 0.75 m × 0.75 m × 0.9 m discretized with cubical macro cells of dimension 0.03 m. The square inlet is 0.15 m on each side, is centered along the back wall at an elevation of 0.675 m from the floor, and provides an inlet flow with a velocity of 2 m/s. Twelve frames are presented in Fig. 23, displaying the free surface from $t = 0.009$ s until $t = 8.763$ s, which is when the fluid level inside the cavity rises just past the top level of the inlet. After impact with the bottom of the tank, which takes place after 0.4 s, the fluid spreads in all directions. A second impact with the downstream wall and a runup effect take place at around 0.51 s, followed immediately by smaller impacts with the lateral walls. The last strong impact takes place with the upstream wall after 0.78 s, followed by a runup event that takes place on the same wall. As long as the fluid level remains below the inlet level ($t = 5.884$ s), a difference in water depth exists between the downstream and upstream walls due to the jet orientation with respect to the main fluid body. After the fluid level reaches the inlet, the oscillations inside the tank are dampened. An interesting physical effect appears after the inlet is totally submerged under the main fluid body, with the formation of an eddy train on each side of the jet ($t = 8.763$ s).

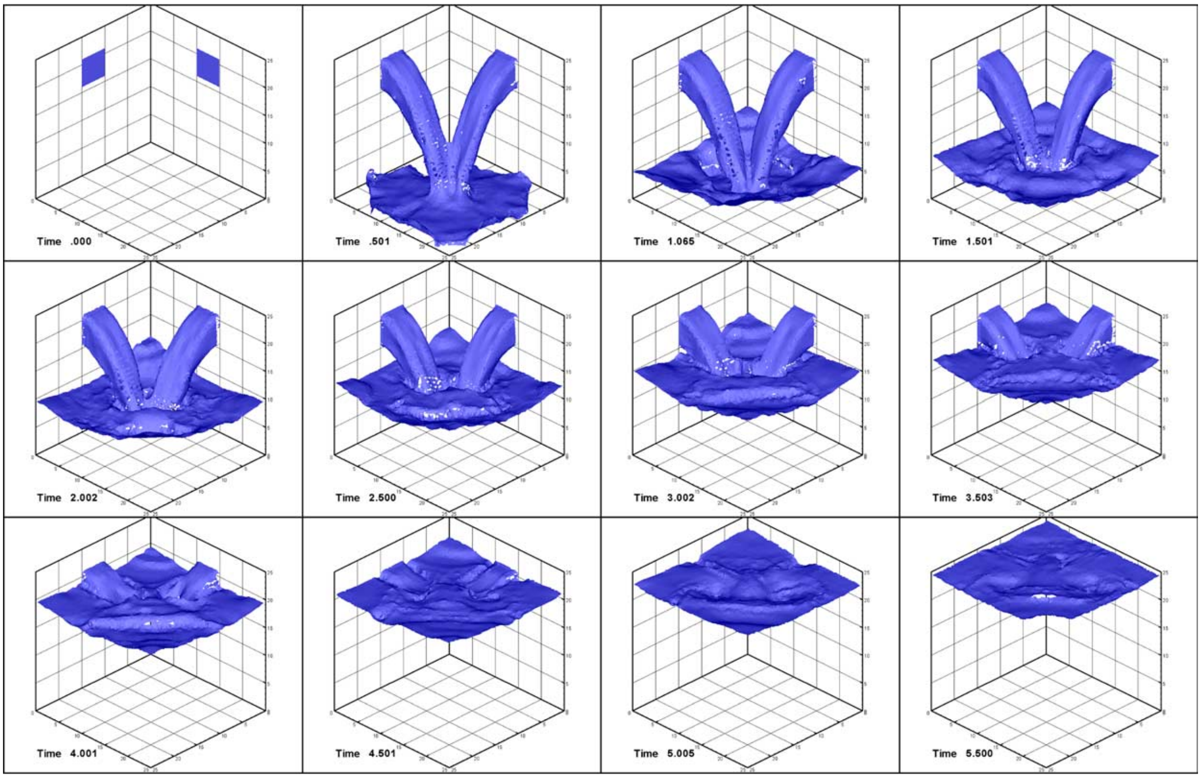


Fig. 24. Simulation of the cavity filling problem with two jets.

The next example adds a second jet to the previous cavity-filling problem. The animation results for this case are again presented in the form of a sequence of still images in Fig. 24. The example is used to highlight another strong capability of the ELMMC method, namely the simulation of fluid–fluid interactions (impact between jets). For this problem, the computational domain consists of a cubical reservoir with dimension 0.625 m, discretized with cubical macro cells of dimension 0.025 m. The square inlets are 0.125 m on each side, and are centered along the back walls at an elevation of 0.4 m from the floor. The inlet flow velocity for each inlet is 1.5 m/s.

Twelve frames are presented in Fig. 24, displaying the free surface from the initial state of the problem at $t = 0$ s until the fluid level inside the reservoir reaches the ceiling at $t = 5.5$ s. A primary impact between the two fluid jets occurs after 0.3 s, and is followed by their coalescence in mid-air. The combined jets impact the bottom cavity after 0.4 s. The speed of the combined jets is much smaller than in the previous simulation, and the runup on the sidewalls is less pronounced. After impact, the fluid spreads in all directions, and secondary impacts occur with lateral walls after $t = 0.501$ s. The impinging (combined) jets for this case emanate from a lower elevation than the single jet in the previous example, and as a result, pierce the pool of water nearly at its center and with a smaller velocity. As a result, the free surface undergoes smaller deformations ($t = 1.065$–2.5 s). Again, the free surface ripples are dampened when the water level reaches the top of inlet. After the water level rises above the inlet, two long eddy trains are generated by the submerged jets, as can seen in the penultimate frame, $t = 5.005$ s.

6. Conclusions

This work presented a new numerical method for handling three-dimensional, incompressible, flows with or without a free surface. Several examples were presented to demonstrate the capabilities of the ELMMC method in simulating flows with severe free surface deformations, including impact with solid boundaries and impact between converging fluid fronts. The validity and consistency of the new method were demonstrated by comparing numerical results with those of a large experimental study. The accuracy of the method was verified through spatial and temporal convergence studies.

The ELMMC method successfully combines the Eulerian Approach for solving the flow variables and the Lagrangian approach for tracking the free surface interface. The use of the Lagrangian surface markers avoids the diffusion (interface thickening) normally associated with the use of Eulerian implements to track the interface as well as the destruction and reconstruction of the interface at each time level. Derived quantities such as surface tension are easily and accurately handled by connecting markers with cubic splines. The ELMMC method is powerful, computationally efficient, robust, and accurate. Its main disadvantage lies primarily in the complexity of programming the various possible cases to handle the computation of the primary variables.

Acknowledgments

P.E.R. gratefully acknowledges that this material is based upon work supported by the National Science Foundation under Grant Nos. CTS-9422762, CMS-9503604, and CMS-9978680. Any opinions, findings, conclusions, or recommendations expressed in this work are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] A.A. Amsden, F.H. Harlow, The SMAC Method: a numerical technique for calculating incompressible fluid flows, LASL Report No. LA-4370, 1970.

[2] V. Armenio, Improved MAC Method (SIMAC) for unsteady high-Reynolds free surface flows, Int. J. Numer. Meth. Fluids 24 (1997) 185.

[3] N. Ashgriz, J.Y. Poo, FLAIR: Flux Line-Segment Model for Advection and Interface Reconstruction, J. Comput. Phys. 92 (1991) 449.

[4] G.K. Batchelor, An Introduction to Fluid Dynamics, Cambridge University Press, Cambridge, 1967.

[5] R. Bidoae, Interaction of free surface flow with solid obstacles and porous media, Ph.D. Dissertation, Mechanical Engineering Department, Southern Methodist University, Dallas, TX, 2000.

[6] M.B. Bush, T.N. Phan, Three dimensional viscous flow with a free surface: flow out of a square die, J. Non-Newtonian Fluid Mech. 18 (1970) 211.

[7] R.K.C. Chan, R.L. Street, A computer study of finite-amplitude water waves, J. Comput. Phys. 6 (1970) 68.

[8] S. Chen, D.B. Johnson, P.E. Raad, Velocity boundary conditions for the simulation of free surface fluid flow, J. Comput. Phys. 116 (1995) 262.

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

[10] A.J. Chorin, A numerical method for solving incompressible viscous flow problems, J. Comput. Phys. 2 (1967) 12.

[11] P.G. Drazin, W.H. Reid, Hydrodynamic Stability, Cambridge University Press, Cambridge, 1981.

[12] D. Fadda, P.E. Raad, S. Chen, D.B. Johnson, Free surface water flow past rigid porous media, in: Proceedings of ASME IMECE, San Francisco, CA, November 12–17, 1995.

[13] J.M. Floryan, H. Rasmussen, Numerical methods for viscous flow with moving boundary, Appl. Mech. Rev. 42 (1989) 322–340.

[14] D. Gueyffier, A. Nadim, R. Scardovelli, S. Zaleski, Volume of fluid interface tracking with smoothed surface stress methods for three dimensional flow, J. Comput. Phys. 152 (1998) 423.

[15] F.H. Harlow, J.E. Welch, Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface, Phys. Fluids 8 (1965) 2182.

[16] X. He, S. Chen, R. Zhang, A Lattice–Boltzmann scheme for incompressible multiphase flow and its application in simulation of Rayleigh–Taylor instability, J. Comput. Phys. 152 (1999) 642.

[17] C.W. Hirt, B.D. Nichols, Volume of Fluid (VOF) method for the dynamics of free boundaries, J. Comput. Phys. 39 (1981) 201.

[18] D.B. Johnson, P.E. Raad, S. Chen, Simulation of impacts of fluid free surfaces with solid boundaries, Int. J. Numer. Meth. Fluids 19 (1994) 153.

[19] D. Juric, G. Tryggvason, A front-tracking method for dendritic solidification, J. Comput. Phys. 123 (1996) 127.

[20] W. Kanok-Nukukchai, B.T. Tam, Structure–fluid interaction model of tuned liquid dampers, Int. J. Numer. Meth. Eng. 46 (1998) 1541.

[21] L.D. Landau, E.M. Lifshitz, Fluid Mechanics, second ed., Pergamon Press, Oxford, 1987.

[22] C.M. Lemos, Wave Breaking – A Numerical Study, Lecture Notes in Engineering, vol. 71, Springer, Berlin, 1992.

[23] C. Lu, Y. He, Nonlinear fluid–structure interaction during two-dimensional water impact, in: Proceedings of the 3rd International Conference on Nonlinear Mechanics, Shanghai, China, 1998, p. 513.

[24] H. Miyata, J. Nishimura, Finite-difference simulation of nonlinear ship waves, J. Fluid Mech. 157 (1985) 327.

[25] D. Munford, J. Shah, Optimal approximation by piecewise smooth functions and associated variational problems, Commun. Pure Appl. Math. 42 (1989) 577.

[26] P.I. Nakayama, N.C. Romeo, Numerical method for almost three-dimensional incompressible fluid flow and a simple internal obstacle treatment, J. Comput. Phys. 8 (1971) 434.

[27] B.D. Nicholas, C. W Hirt, Calculating three-dimensional free surface flows in a vicinity of submerged and exposed structures, J. Comput. Phys. 12 (1973) 234.

[28] W. Noh, P. Woodward, SLIC (Simple Line Interface Calculation), in: A.I. Vande Vooren, P.J. Zandbergen (Eds.), Proceedings of the 5th International Conference on Fluid Dynamics, Lecture Notes in Physics, vol. 59, Springer, Berlin, 1976, p. 330.

[29] S. Osher, J.A. Sethian, Front propagating with curvature-dependent speed: algorithm based on Hamilton–Jacobi formulations, J. Comput. Phys. 79 (1988) 12.

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

[31] P.E. Raad, S. Chen, D.B. Johnson, The introduction of micro cells to treat pressure in free surface fluid flow problems, J. Fluids Eng. 117 (1995) 683.

[32] B. Ramaswamy, M. Kawahara, Arbitrary Lagrangian–Eulerian finite element method for unsteady, convective, incompressible viscous free surface fluid flow, Int. J. Numer. Meth. Fluids 7 (1987) 1053.

[33] W.J. Rider, D.B. Kothe, Stretching and tearing interface tracking methods, AIAA Paper 95 (1995) 17.

[34] M. Rudman, A volume tracking method for incompressible multifluid flows with large density variations, Int. J. Numer. Meth. Fluids 28 (1998) 357.

[35] S. Ruuth, B. Merriman, S. Osher, A fixed grid for capturing the motion of self-intersecting interface and related PDEs, J. Comput. Phys. 163 (2000) 21.

[36] A. Sabau, Numerical methods for fluid–solid interaction problems with moving boundaries, Ph.D. Dissertation, Mechanical Engineering Department, Southern Methodist University, Dallas, TX, 1996.

[37] R. Scardovelli, S. Zaleski, Direct numerical simulation of free surface and interaction flow, Annu. Rev. Fluid Mech. 31 (1999) 567.

[38] H.M. Soner, L. Ambrosio, Flow by mean curvature of surfaces of any co-dimension, Prog. Nonlinear Diff. Eq. Their Appl. 25 (1996) 135.

[39] M.F. Tome, S. McKee, GENSMAC: a computational marker and cell method for free surface flows in general domains, J. Comput. Phys. 110 (1994) 171.

[40] M.F. Tome, A.C. Filho, J.A. Cuminato, N. Mangiavacchi, S. McKee, GENSMAC3D: a numerical method for solving unsteady three-dimensional free surface flows, Int. J. Numer. Meth. Fluids 37 (2001) 747.

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

[42] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, Y.-J. Jan, A front-tracking method for the computations of multiphase flow, J. Comput. Phys. 169 (2001) 708.

[43] T. Tsukamoto, K. Anzai, E. Niyama, Tracing technique for free surface of fluid flow by FVM method using an irregular grid, Trans. Jpn. Soc. Mech. Eng. B 63 (1997) 1961.

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

[45] J.A. Viecelli, A computing method for incompressible flows bounded by moving walls, J. Comput. Phys. 8 (1971) 119.