

AN EULERIAN FINITE ELEMENT METHOD FOR TIME-DEPENDENT FREE SURFACE PROBLEMS IN HYDRODYNAMICS

TSUKASA NAKAYAMA AND MINEO MORI

Department of Precision Mechanics, Chuo University, 13-27, Kasuga 1-chome, Bunkyo-ku, Tokyo 112, Japan

SUMMARY

A numerical method based on the finite element method is presented for simulating the two-dimensional transient motion of a viscous liquid with free surfaces. For ease of numerical treatment of the free surface expressed by a multiple-valued function, the marker particle method is employed. Numerous virtual particles are spread over all regions occupied by liquid. They move about on a fixed finite element mesh with the liquid velocity at their positions. These particles contribute nothing to the dynamics of the liquid and only serve as markers of liquid regions. The velocity field within liquid regions is calculated by solving the Navier–Stokes equations and the equation of continuity by the finite element method based on quadrilateral elements. A detailed discussion is given of the methodological problems arising in the implementation of the marker particle method on an unstructured finite element mesh and of the solutions to these problems. The proposed method is demonstrated on three sample problems: the broken dam problem, the impact of a falling liquid drop on a still liquid and the entry of a rigid block into water. Good agreement has been obtained in the comparison of the present numerical results with available experimental data.

KEY WORDS: free-surface flow; viscous incompressible fluid; finite element method; marker particle method

1. INTRODUCTION

The primary interest of the present work is to develop an effective method for numerically simulating the transient motion of a liquid with free surfaces. There are two approaches in the numerical analysis of free surface flow problems: the Lagrangian method and the Eulerian method. In the former method, the liquid regions are subdivided into finite difference or finite element meshes and each cell or element is moved and deformed according to the liquid motion. Therefore the mesh zone is identified with the liquid region for all time. The governing equations of flow are simple and easy to solve in the Lagrangian formulation, since the momentum equations have no non-linear advection terms. However, a rezoning or remeshing procedure is needed to avoid the serious distortion of cells and one must have a fast and fully automatic mesh generator for this procedure. Recently the arbitrary Lagrangian–Eulerian (ALE) method has often been used.^{1–5} In this method, to avoid the rapid distortion of cells, each cell can be moved with an arbitrary velocity which is independent of the liquid velocity. However, rezoning is still needed even in the ALE method.

On the other hand, a mesh remains fixed in the Eulerian method and liquid regions change in shape and location on the mesh. Therefore the identity between the mesh zone and the liquid region is not maintained. Then an additional method is needed to recognize the area occupied by liquid in the Eulerian formulation. Several methods have been used for this purpose, such as the height function method,⁶ the volume-of-fluid (VOF) method⁷ and the marker particle method.⁸

In the height function method a free surface is represented by its vertical height measured from a reference line and a liquid region is recognized as the region between the free surface and the base of a solution domain. This method is very easy to implement and requires only a one-dimensional storage array to record the discrete value of the free surface. However, since the surface height is expressed by a single-valued function of position, the method does not work for free surfaces expressed by multiple-valued functions, as exemplified by bubbles, liquid droplets, overturning water waves and so on.

The VOF method uses an auxiliary function F whose value is unity at any point in the liquid and zero elsewhere. The time evolution of F is governed by the pure advection equation

$$\frac{\partial F}{\partial t} + u_i F_{,i} = 0,$$

where u_i is the velocity component of the liquid. Usually the function F is discretized so as to be uniform within each cell and the value of F in a cell represents the fractional volume of liquid in the cell. A unit value of F corresponds to a cell full of liquid, while a zero value corresponds to a cell containing no liquid. A value of F between zero and unity indicates that a cell contains a free surface. Although the storage requirements increase in the VOF method in comparison with the height function method, only one storage word is required for each cell. The VOF method can be applied to the dynamics of multiple-valued surfaces.

The marker particle method uses virtual particles to represent liquid regions. The particles are spread over all regions occupied by liquid. They have no quantifiable properties such as mass or energy and serve only as 'markers' showing liquid regions. The computational procedure is very simple. First the governing flow equations are solved on the cells containing particles and the velocity field in the liquid is determined. Next each particle is moved according to the liquid velocity at its position. This procedure is repeated by advancing the time by Δt . The location and profile of a free surface are recognized by observing a computer plot of the marker distribution. Thus the marker particle method eliminates logic problems associated with the calculation of free surface locations and can be easily applied to problems with multiple-valued free boundaries. Furthermore, it is readily extensible to three-dimensional computations. In comparison with the two previous methods, however, storage requirements increase considerably because of the increase in the number of particle co-ordinates that must be recorded. Additional computational time is also required to move all particles. However, these requirements can be tolerated because of the recent increase in performance of digital computers.

If the Lagrangian approach is adopted, one must prepare an automatic mesh generator for the rezoning of a solution domain. Although several algorithms of mesh generation have been proposed for restricted purposes such as the hot-forming process⁹ and metal-casting flow,¹⁰ no fully automatic and general-purpose mesh generator can be found to date. Furthermore, the rezoning process will be more complicated in three-dimensional computations. Therefore we will employ the Eulerian approach for the present work. Among various techniques representing liquid regions on an Eulerian mesh, some of which are briefly reviewed above, we have chosen the marker particle method. Although this method has some disadvantages such as the vague definition of free surface and the resultant difficulty in including surface tension effects, it has attractive advantages such as simplicity of the algorithm, the extensibility to three-dimensional computations and wide applicability.

A successful example of the marker particle method can be found in the marker-and-cell (MAC) method.⁸ It is based on the finite difference method and fundamentally the solution domain and obstacles in the domain are required to be rectangular. In the subsequent developments of the original MAC method by many researchers,¹¹⁻¹⁴ Viacelli¹⁴ presented the extended MAC method, named ABMAC, which is applicable to the calculation of free surface flow in arbitrarily shaped domains. However, the treatment of cells in the vicinity of curved wall boundaries and the calculation of momentum and pressure in those cells are troublesome in ABMAC. Furthermore, it employs the

inviscid boundary condition on a free surface. Here we propose a new technique by combining the marker particle method with the finite element method (FEM). By employing the FEM, it becomes quite easy to generalize the treatment of a curved wall boundary and to introduce the viscous stress condition on a free surface. In the present paper we describe the details of the proposed solution technique. Although the present method is more generalized than ABMAC, the present work is influenced by ABMAC in the basic strategy of computation.

2. DESCRIPTION OF A FREE SURFACE PROBLEM

As an example of the mathematical formulation of a free surface problem we consider the two-dimensional motion of a liquid in a container as illustrated in Figure 1. Let Ω be the liquid region in the container. This region is bounded by two types of boundaries: a free surface Γ_1 and a wetted part of the wall of the container, Γ_2 . The flow of liquid is assumed to be laminar. Then the liquid motion under gravity is governed by the Navier-Stokes equations

$$\frac{\partial u_i}{\partial t} + u_j u_{i,j} = \sigma_{ij,j} + f_i \quad \text{in } \Omega \tag{1}$$

and the equation of continuity

$$u_{i,i} = 0 \quad \text{in } \Omega, \tag{2}$$

where t is the time, x_i ($i = 1, 2$) are rectangular Cartesian co-ordinates, u_i is the velocity component in the x_i -direction, f_i is the x_i -component of the gravitational force per unit mass of liquid and σ_{ij} is the stress tensor. For a Newtonian fluid the following constitutive equation applies:

$$\sigma_{ij} = -p\delta_{ij} + v(u_{i,j} + u_{j,i}), \tag{3}$$

where p is the pressure per unit density and v is the coefficient of kinematic viscosity.

On the free surface boundary the equilibrium condition of stress,

$$\sigma_{ij}n_j = 0 \quad \text{on } \Gamma_1, \tag{4}$$

is imposed, where n_i is the x_i -component of the unit outward vector normal to the boundary. In the derivation of equation (4) the pressure of the gas in the container is assumed to be zero and the surface tension of the liquid is neglected. The wall of the container is assumed to be impermeable. Then the following free slip condition is applied to the boundary Γ_2 :

$$u_i n_i = \sigma_{ij}n_i t_j = 0 \quad \text{on } \Gamma_2, \tag{5}$$

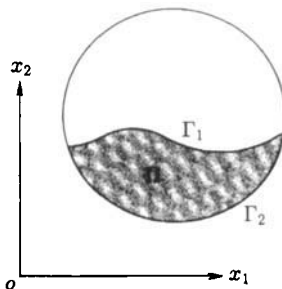


Figure 1. Liquid with a free surface in a two-dimensional container

where t_i is the x_i -component of the unit vector tangential to the boundary. For the case of a viscous fluid the no-slip condition may be desirable. However, owing to the finite mesh size, an unnecessarily large boundary layer is created using a numerical no-slip condition. Therefore the free slip condition is preferable for the present calculation.

The initial condition for equation (1) is given by specifying the velocity field which satisfies equation (2) at the initial state.

3. FINITE ELEMENT FORMULATION

3.1. Spatial and temporal discretization

Equations (1) and (2) are discretized in space by using the Galerkin FEM based on quadrilateral elements. The velocity is defined at four vertices of an element and its distribution in the element is approximated by a bilinear polynomial. The pressure node is located at the centroid of an element and the pressure is assumed to be uniform in the element. Thus the semidiscretized equations are derived as

$$\bar{\mathbf{M}}\dot{\mathbf{U}} + [\mathbf{A}(\mathbf{U}) + \mathbf{K}]\mathbf{U} - \mathbf{H}\mathbf{P} - \mathbf{F} = \mathbf{O}, \quad (6)$$

$$\mathbf{h}_e^T \mathbf{u}_e = 0 \quad (e = 1, 2, \dots, NE), \quad (7)$$

where \mathbf{U} and \mathbf{P} are vectors of nodal values of velocity and pressure respectively; $\bar{\mathbf{M}}$ is the lumped mass matrix, \mathbf{A} , \mathbf{K} and \mathbf{H} are advection, diffusion and pressure gradient matrices respectively, \mathbf{F} is the vector of gravitational force and \mathbf{O} is a null vector. The superposed dot of \mathbf{U} denotes Eulerian time differentiation. The area integration for evaluating the matrices $\bar{\mathbf{M}}$, \mathbf{A} , \mathbf{K} and \mathbf{H} is carried out analytically.¹⁵ The boundary condition (4) is taken into account as a natural boundary condition when the weak form of equation (1) is derived. The numerical way of introducing the free slip condition (5) into equation (6) can be found in Reference 16. Equation (7) represents the local mass conservation in an element Ω_e . The subscript e refers to the element Ω_e and NE denotes the total number of elements. The row vector \mathbf{h}_e^T corresponds to the divergence operator and \mathbf{u}_e denotes the vector of nodal velocity of the element Ω_e .

The time axis is divided into a set of segments with short and uniform length Δt and the calculation is progressed by advancing the time by Δt . In a typical time interval between $t^n = n\Delta t$ and $t^{n+1} = (n+1)\Delta t$, equation (6) is discretized in time as

$$\bar{\mathbf{M}}\mathbf{U}^{n+1} - \bar{\mathbf{M}}\mathbf{U}^n + \Delta t\{[\mathbf{A}(\mathbf{U}^n) + \mathbf{K}]\mathbf{U}^n - \mathbf{H}\mathbf{P}^{n+1} - \mathbf{F}\} = \mathbf{O}, \quad (8)$$

where superscripts n and $n+1$ refer to the time instants t^n and t^{n+1} respectively. The equation of continuity (7) is rewritten as

$$\mathbf{h}_e^T \mathbf{u}_e^{n+1} = 0 \quad (e = 1, 2, \dots, NE). \quad (9)$$

For convenience in the subsequent description, equation (8) is simplified as

$$\mathbf{U}^{n+1} - \Delta t \bar{\mathbf{M}}^{-1} \{\mathbf{A}(\mathbf{U}^n) + \mathbf{K}\}\mathbf{U}^n - \mathbf{Q}^n = \mathbf{O}, \quad (10)$$

where

$$\mathbf{Q}^n = \mathbf{U}^n - \Delta t \bar{\mathbf{M}}^{-1} \{\mathbf{A}(\mathbf{U}^n) + \mathbf{K}\}\mathbf{U}^n - \mathbf{F}. \quad (11)$$

Our next task is to calculate \mathbf{U}^{n+1} and \mathbf{P}^{n+1} satisfying equations (9) and (10) with the known \mathbf{U}^n and \mathbf{P}^n . For this purpose a new type of element-by-element (EBE) time-stepping method is constructed by modifying Chorin's method.¹⁷ The details are described in the next subsection.

3.2. EBE time-stepping procedure.

The advancement of the flow field variables in a time step is carried out in two stages. In the first stage the velocity is advanced by using the previous state of the flow field. However, this explicit time advancement does not necessarily lead to a velocity field with zero divergence. Thus in the second stage the velocity is corrected so as to ensure mass conservation. This correction is made by adjusting the pressure of each element in such a way that the local divergence of velocity in an element should vanish. A change of velocity in an element will affect its neighbouring elements and this pressure adjustment must be performed in an iterative manner until all elements attain simultaneously zero divergence of velocity.

First the velocity at time t^{n+1} is estimated by

$$\tilde{\mathbf{U}}^{n+1} = \Delta t \bar{\mathbf{M}}^{-1} \mathbf{H} \mathbf{P}^n + \mathbf{Q}^n, \quad (12)$$

which is obtained by replacing \mathbf{P}^{n+1} with \mathbf{P}^n in equation (10). Since the velocity thus calculated does not necessarily satisfy equation (9), it is labelled by a tilde.

Next the above tilded velocity is corrected by iterative calculations so as to ensure mass conservation. Consider the $(k+1)$ th stage of iteration in which the $(k+1)$ th approximations of \mathbf{U}^{n+1} and \mathbf{P}^{n+1} are calculated using the known k th approximations. The $(k+1)$ th stage starts from the calculation of the local velocity divergence $D_e^{(k)}$ per unit area in an element Ω_e . It is evaluated by

$$D_e^{(k)} = \frac{1}{A_e} \mathbf{h}_e^T \mathbf{u}_e^{n+1,(k)}, \quad (13)$$

where A_e is the area of the element and the superscript (k) denotes the k th approximation. The zeroth approximations of velocity and pressure are given as $\mathbf{u}_e^{n+1,(0)} = \tilde{\mathbf{u}}_e^{n+1}$ and $p_e^{n+1,(0)} = p_e^n$ respectively. For simplicity of expression the superscript $n+1$ showing the time level will be omitted in the following.

If the magnitude of $|D_e^{(k)}|$ is less than some prescribed small value ε , the flow is considered locally incompressible and no adjustment is made. If the magnitude exceeds ε , the pressure in Ω_e is adjusted as

$$p_e^{(k+1)} = p_e^{(k)} + \Delta p_e^{(k)}. \quad (14)$$

The pressure change $\Delta p_e^{(k)}$ is computed by

$$\Delta p_e^{(k)} = -\lambda_e D_e^{(k)}, \quad (15)$$

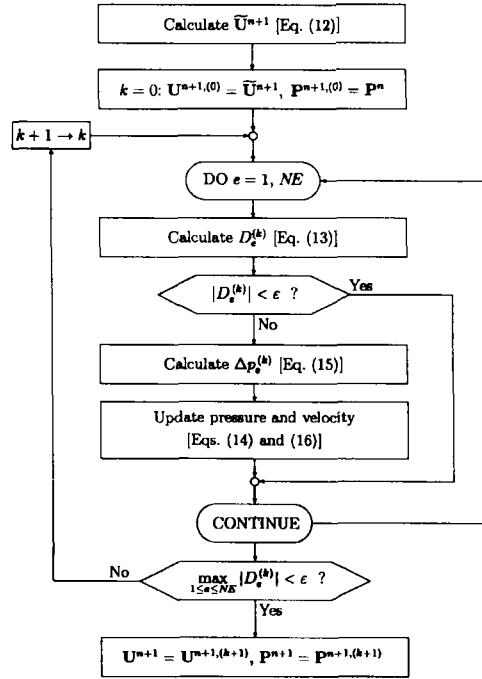
where λ_e is a positive coefficient which depends on the element size and Δt . The expression for λ_e is derived later in this subsection. The pressure change affects the local velocity field in Ω_e and the nodal value of velocity is updated as

$$\mathbf{u}_e^{(k+1)} = \mathbf{u}_e^{(k)} + \Delta t \bar{\mathbf{m}}_e^{-1} \mathbf{h}_e \Delta p_e^{(k)}, \quad (16)$$

where $\bar{\mathbf{m}}_e^{-1}$ is an 8×8 submatrix of $\bar{\mathbf{M}}^{-1}$ whose elements correspond to four nodes of Ω_e .

A series of calculations given by equations (13)–(16) in the $(k+1)$ th stage proceeds element-by-element and the iterative process is repeated until no element has a magnitude of $|D_e|$ greater than ε . Thus the converged values of velocity and pressure are stored as the values at time t^{n+1} . The time-stepping procedure in the interval between t^n and t^{n+1} is summarized in Figure 2. In general the size of ε varies according to the problem to be solved. We have used the value of 10^{-4} for ε in the present work.

It should be noted here that $\mathbf{u}_e^{(k+1)}$ is affected not only by $\Delta p_e^{(k)}$ but also by the pressure change in the elements adjacent to Ω_e . Therefore, if a node is shared by four elements, the velocity at that node will be corrected four times during one iteration.

Figure 2. Time-stepping procedure in interval between t^n and t^{n+1}

The expression for the coefficient λ_e can be derived as follows. Multiplication by \mathbf{h}_e^T/A_e on both sides of equation (16) yields

$$D_e^{(k+1)} = D_e^{(k)} + \frac{\Delta t}{A_e} \mathbf{h}_e^T \bar{\mathbf{m}}_e^{-1} \mathbf{h}_e \Delta p_e^{(k)}. \quad (17)$$

The pressure adjustment by $\Delta p_e^{(k)}$ is made so that $D_e^{(k+1)}$ should vanish. Then, putting $D_e^{(k+1)}$ equal to zero in the above equation, we obtain the relation between $\Delta p_e^{(k)}$ and $D_e^{(k)}$ as

$$\Delta p_e^{(k)} = -\frac{A_e}{\Delta t \mathbf{h}_e^T \bar{\mathbf{m}}_e^{-1} \mathbf{h}_e} D_e^{(k)}. \quad (18)$$

Thus, by identifying the coefficient of $D_e^{(k)}$ in equation (15) with that in equation (18), the expression

$$\lambda_e = \frac{A_e}{\Delta t \mathbf{h}_e^T \bar{\mathbf{m}}_e^{-1} \mathbf{h}_e} \quad (19)$$

is obtained.

3.3. Stability analysis of the iterative calculation

In the finite difference calculations^{14,18} the right-hand side of equation (15) is multiplied by a relaxation parameter ω to accelerate the iterative calculation. In this case ω may not exceed two and the value of 1.7 is commonly used for ω , though it is occasionally changed according to the problem to be solved. In this subsection, we consider a permissible range of ω for the present finite element calculation through the stability analysis of the iterative procedure of the previous subsection.

First we show that the iterative calculation between the velocity correction and the pressure advancement is equivalent to the time-marching calculation for obtaining the steady state solution of an evolution equation of pressure. The differential form corresponding to equation (16) is expressed as

$$u_i^{n+1,(k+1)} = u_i^{n+1,(k)} - \Delta t (\Delta p)_{,i}^{(k)} \quad (20)$$

and the zeroth approximation $u_i^{n+1,(0)}$ is given by

$$u_i^{n+1,(0)} = \tilde{u}_i^{n+1} = q_i^n - \Delta t p_{,i}^n. \quad (21)$$

Here q_i^n is defined as

$$q_i^n = u_i^n - \Delta t [u_j^n u_{i,j}^n - v(u_{i,j}^n + u_{,i}^n)_j - f_i] \quad (22)$$

and is kept unchanged during the iteration. The recursive use of equation (20) under the condition (21) yields

$$u_i^{n+1,(k)} = q_i^n - \Delta t p_{,i}^{n+1,(k)}. \quad (23)$$

At this time the relation

$$p^{n+1,(k)} = p^n + \sum_{l=0}^{k-1} (\Delta p)^{(l)} \quad (24)$$

is used. By substituting relation (23) into the differential expression

$$p^{n+1,(k+1)} - p^{n+1,(k)} = -\omega \lambda u_{,i}^{n+1,(k)} \quad (25)$$

corresponding to equation (15), we obtain

$$p^{n+1,(k+1)} - p^{n+1,(k)} = \omega \lambda \Delta t p_{,ii}^{n+1,(k)} - \omega \lambda q_{i,i}^n \quad (26)$$

or

$$\frac{p^{n+1,(k+1)} - p^{n+1,(k)}}{\Delta \tau} - p_{,ii}^{n+1,(k)} = -\frac{1}{\Delta t} q_{i,i}^n. \quad (27)$$

Here ω is the aforementioned relaxation parameter and $\Delta \tau = \omega \lambda \Delta t$. If $\Delta \tau$ is considered as an increment on a pseudotime axis τ , equation (27) can be interpreted as a temporally discretized equation of the following evolution equation of pressure:

$$\frac{\partial p^{n+1}}{\partial \tau} - p_{,ii}^{n+1} = -\frac{1}{\Delta t} q_{i,i}^n. \quad (28)$$

The advancement of τ corresponds to an increase in the iteration number k . By continued iteration the value of $\partial p^{n+1} / \partial \tau$ decreases and one approaches the solution of the pressure Poisson equation,

$$p_{,ii}^{n+1} = \frac{1}{\Delta t} q_{i,i}^n. \quad (29)$$

The stability of the iterative calculation for pressure updating is equivalent to that of equation (28). By applying the von Neumann stability analysis to the finite element representation of equation (28), we can obtain the result that one must choose $\Delta \tau$ as

$$\Delta \tau \leq \frac{2(\Delta x_1 \cdot \Delta x_2)^2}{(\Delta x_1)^2 + (\Delta x_2)^2} \quad (30)$$

on a uniform mesh pattern composed of rectangular elements of size $\Delta x_1 \times \Delta x_2$. Since $\Delta \tau = \omega \lambda_e \Delta t$ in the discrete form of equation (28) and λ_e is given by equation (19), we can obtain the condition $\omega \leq 2$ from the inequality (30). This is the same result as in the finite difference formulation. In general the value of ω depends on the problem to be solved. However, to save the effort of finding a suitable value of ω according to the problem ω is fixed as unity in the present work.

4. COMPUTATIONAL PROCEDURE

4.1. Algorithm

The computational domain does not necessarily coincide with the region occupied by liquid in the present method. The computational domain is set so as to have enough extent to allow liquid to move around in the domain. In Figure 1, for example, the whole of the inside of the container is chosen as the computational domain. The computational domain is subdivided into finite elements and the mesh remains fixed for all time. A lot of massless particles are distributed over all liquid regions. In the numerical examples shown in the next section, for example, particles are arranged at the rate of 14 or 18 per element at the initial state. Each particle is moved with the liquid velocity at its position. The particle contributes nothing to the dynamics of the liquid and only serves as a marker to represent a liquid region. We call such a virtual particle a marker.

Consider a certain time instant t^n and suppose that the marker distribution and the velocity and pressure fields at that time are known. Then the simulation procedure is summarized as follows.

1. As shown in Figure 3, all the finite elements are sorted into three types: empty element, surface element and fluid element. An empty element contains no markers. Surface and fluid elements contain several markers. A surface element adjoins one or more empty elements, while a fluid element is adjacent to no empty elements. The region composed of a group of surface and fluid elements is regarded as a liquid region and the polygonal line composed of the common sides of empty and surface elements represents an approximate free surface.
2. The governing flow equations (1) and (2) are discretized on the surface and fluid elements to yield the matrix equations (6) and (7). By solving these equations, the velocity field is obtained.
3. The velocity of a marker is calculated at its position by interpolating the nodal velocity of the element containing the marker. The marker is moved according to

$$x_i^{n+1} = x_i^n + u_i^{n+1} \Delta t, \quad (31)$$

where (x_1^n, x_2^n) is the current marker position, (x_1^{n+1}, x_2^{n+1}) is the updated marker position and (u_1^{n+1}, u_2^{n+1}) is the marker velocity.

After the time has been advanced by Δt , the procedure is repeated from step 1.

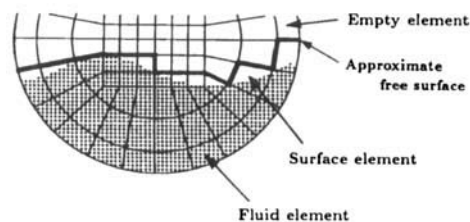


Figure 3. Classification of elements and definition of an approximate free surface

4.2. Search of marker position

To evaluate the velocity of a marker by interpolation, we have to find the element in which the marker is located. If the computational domain is subdivided into a uniform mesh composed of rectangular elements, the location of a marker with respect to the elements can be easily found. In the MAC method, for example, the integer numbers i and j calculated by

$$i = \text{integer part of } \left(\frac{x_1}{\Delta x_1} + 2 \right), \quad j = \text{integer part of } \left(\frac{x_2}{\Delta x_2} + 2 \right)$$

inform us that the marker with co-ordinates (x_1, x_2) is in the cell ij . Here Δx_1 and Δx_2 are the cell width and height respectively. On an unstructured mesh, however, the search of marker position is not so easy.

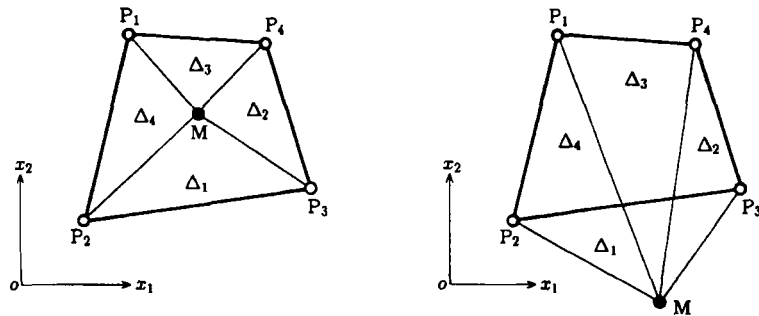
The method employed here is the application of the area co-ordinates used in the finite element formulation on triangular elements. Consider a typical quadrilateral element $P_1P_2P_3P_4$ and a marker M as shown in Figure 4 and let (x_1, x_2) with superscripts $M, 1, 2, 3$ and 4 be the rectangular Cartesian co-ordinates of the marker and four vertices of the element respectively. Then the four quantities defined by

$$\Delta_i = \begin{vmatrix} 1 & x_1^M & x_2^M \\ 1 & x_1^i & x_2^i \\ 1 & x_1^k & x_2^k \end{vmatrix}, \quad (i, j, k) = (1, 2, 3), (2, 3, 4), (3, 4, 1), (4, 1, 2), \quad (32)$$

are calculated and their signs are checked.

If all Δ_i ($i = 1, 2, 3, 4$) take positive values, the marker is judged to exist in the element as shown in Figure 4(a). In this case the value of Δ_i is equal to twice the area of the triangle ΔMP_jP_k . If the marker exists on the side of the element, one of the four quantities becomes zero. On the other hand, equation (32) produces a negative value for at least one quantity when the marker lies outside the element. In the case shown in Figure 4(b), for example, Δ_1 takes a negative value. Then one must repeat the calculation of equation (32) in other elements until all Δ_i take non-negative values.

To accelerate the search process, it is helpful to store the element number thus found against the marker number. Since a marker cannot travel more than one element size in one time step, the marker will be expected to remain in the previous element or to move to an element among eight elements adjacent to the previous element. Therefore it is sufficient to check nine elements at most per marker.



(a) Marker M in the element

(b) Marker M lying outside the element

Figure 4. Search of marker position by checking signs of Δ_i ($i = 1, 2, 3, 4$)

4.3. Velocity adjustment in surface elements

Consider a certain empty element adjacent to surface elements at time t^n and suppose that several markers enter the empty element during the time interval between t^n and t^{n+1} . The typical situation is illustrated in Figure 5. The element receiving markers anew (a hatched element in Figure 5) is reclassified as a surface element and contributes to the assemblage of the matrix equations (6) and (7) at time t^{n+1} . Such an element often has nodes at which the liquid velocity is not yet determined and consequently remains zero. Such a node is referred to as 'a newcomer node' hereafter and is shown by an open circle in Figure 5. Since the fluid is in motion, all the nodes on surface elements should have a non-zero value of velocity. Therefore a process to assign a non-zero value of velocity to a newcomer node is needed before the calculation of the tilded velocity by equation (12). The process of velocity assignment is summarized as follows.

1. Calculate the average velocity of markers in each surface element with newcomer nodes.
2. Estimate the velocity at a newcomer node by averaging the element velocity calculated in Step 1 with an area-weighting scheme. Let us consider a sample case as shown in Figure 6, where a newcomer node P belongs to two surface elements Ω_a and Ω_b . In these elements the average values of marker velocity, \bar{u}_i^a and \bar{u}_i^b , are computed, where superscripts a and b refer to the elements Ω_a and Ω_b , respectively. Then the estimated velocity at P, u_i^P , is calculated by

$$u_i^P = \frac{\bar{u}_i^a A_a + \bar{u}_i^b A_b}{A_a + A_b}, \quad (33)$$

where A_a , for example, denotes the area of the element Ω_a .

3. Correct the nodal velocity estimated in Step 2 so that mass conservation is ensured in each surface element. This correction is made in the same iterative manner as described in Section 3.2. The following series of calculations is done in each surface element with newcomer nodes and is repeated until local conservation of mass is attained:
 - (a) calculate the local divergence of velocity by equation (13)
 - (b) calculate the pressure change by equation (18)
 - (c) update the element pressure by equation (14)
 - (d) correct the nodal velocity by equation (16).

The velocity correction in (d) is made only at newcomer nodes and velocities at other nodes remain unchanged during the computation of (a)–(d).

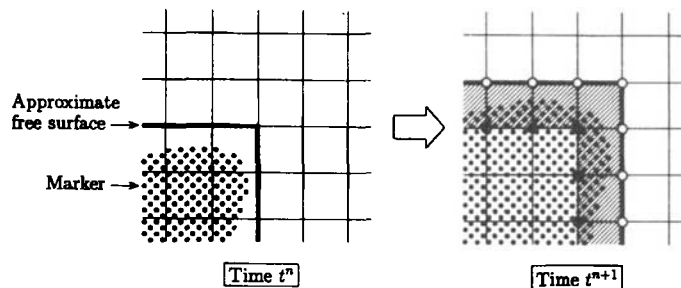


Figure 5. Appearance of newcomer nodes due to marker movement

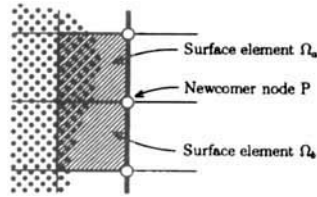


Figure 6. Estimation of velocity at a newcomer node

5. NUMERICAL EXAMPLES

5.1. A broken dam problem

To verify the computational accuracy of the proposed method, the well-known broken dam problem is analysed and the numerical results are compared with available experimental data. A rectangular column of water is confined between two vertical walls and is at rest as shown in Figure 7. It is our task to calculate the collapse of the column under gravity after sudden removal of the right wall. The calculation is carried out for two cases with different relative column heights $b/a = 1$ and 2, where a and b are the column width and height respectively. The rectangular computational domain of size $6a \times 3.75a$ is subdivided into a uniform mesh of 96 elements in the horizontal direction and 60 elements in the vertical direction. The initial marker density is 18 per element. The variables used in the computation are all non-dimensionalized by choosing a as reference length and \sqrt{ga} as reference velocity, where g is the gravitational acceleration. The non-dimensional viscous coefficient $\nu/[a\sqrt{ga}]$, namely the reciprocal of the Reynolds number, is 3.05×10^{-5} and the non-dimensional time increment $\Delta t\sqrt{g/a}$ is taken as 10^{-3} .

Figure 8 shows the computer plots of marker distributions and pressure contours at different time instants. The series of marker distributions shows a smooth motion of the water. In the pressure contours the top curve corresponds to $p/ga = 0.2$ and the contour increment is 0.1. Since the pressure is discretized to be constant in an element, it is hard to plot the continuous contour line of zero pressure.

In Section 4.3 the adjustment procedure for the velocity at a newcomer node of a surface element has been described. The calculation in Figure 8 includes this adjustment procedure. On the other hand, the computed result without velocity adjustment is shown in Figure 9. In this case the velocity at a newcomer node remains zero until the tilded velocity is calculated by equation (12). As a result, the computed marker distributions in Figure 9 show a curious behaviour of the water: the water front is rolled up as if there were a ghost barrier ahead of it. Therefore we might conclude that the procedure

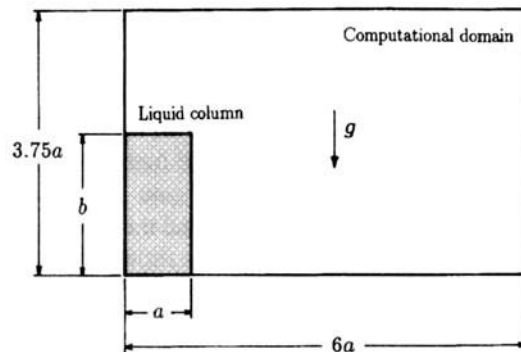


Figure 7. Computational model for broken dam problem

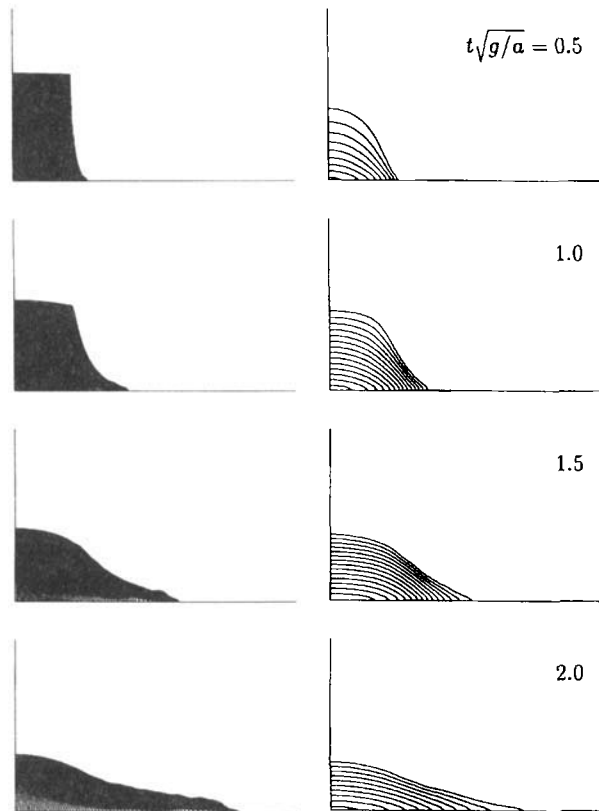


Figure 8. Marker distributions (left) and pressure contours (right) at different time instants

for velocity adjustment at newcomer nodes is important in the present marker particle method. This conclusion may hold true for other Eulerian methods, e.g. the VOF method, and the procedure described in Section 4.3 is also applicable to those Eulerian methods when they are used together with the FEM.

Figure 10 shows the time histories of the water front location $\zeta(t)$ and the water column height $\eta(t)$. The open circles denote the experimental data of Martin and Moyce¹⁹ and the full curves show the present numerical results. In Figure 10(b) the numerical results obtained on the coarser mesh of 48×30 elements are also shown by broken curves. The difference in the computed results between the fine and coarse meshes is not very significant. The present numerical calculation yields satisfactory results which are very close to the experimental data.

5.2. Splash of a falling liquid drop

The next example is the impact of a liquid drop on a still liquid. A circular liquid drop falls towards the free surface of a liquid in a container as shown in Figure 11. Both liquids are assumed to have common material properties. The density is 10^3 kg m^{-3} and the viscous coefficient is $5 \times 10^{-2} \text{ Pa s}$. The computational region is subdivided into elements as shown in Figure 12. The total number of markers used is 23,434 and the initial marker density is 14 per element. The initial falling speed of the drop is 3.2 m s^{-1} and the time increment is set as 10^{-5} s .

Figure 13 shows a series of marker distributions and corresponding pressure contours. The minimum pressure of the contour is 50 Pa and its increment is 50 Pa. In the last plot of the figure we

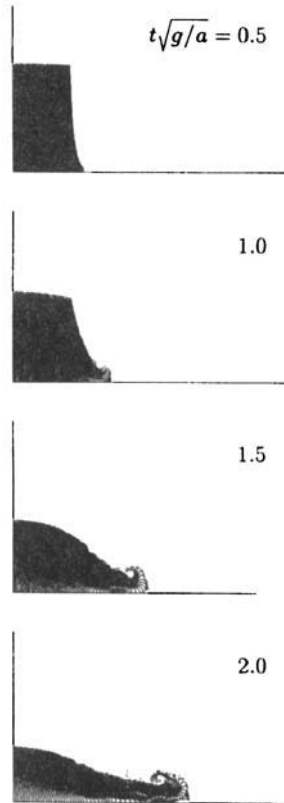


Figure 9. Marker distributions computed without velocity adjustment at newcomer nodes

notice the collision of some markers with the side walls. They show a tendency to move along the wall after the collision. In laboratory experiments, however, the liquid spray will rebound from the wall. Such a rebound cannot be simulated in the present formulation, since the normal component of liquid velocity is fixed to be zero on the wall for all time. Therefore a detailed investigation will be needed for the mathematical modelling of the interaction between a liquid spray and a solid wall.

5.3. Entry of a rigid block into water

The third example concerns the low-velocity penetration of a falling rigid block into still water. This problem is closely related to ship slamming, seaplane landing, water entry of flying objects and so on.

A rectangular block is used for the present computation. An illustrative drawing of the water entry of the block is shown in Figure 14. The block moves downwards so that its axis of symmetry is kept vertical. Since the flow pattern of the water becomes symmetrical, we shall consider hereafter only the right half of the liquid region. The rectangular Cartesian co-ordinate system shown in the figure is a moving one fixed to the block. The origin o is located at the centre of the impact surface of the block and the x_2 -axis coincides with the axis of symmetry and is directed upwards. The liquid region Ω is surrounded by four boundaries: the free surface Γ_1 , the block surface and axis of symmetry, Γ_2 , and the artificial boundaries Γ_3 and Γ_4 . The base boundary Γ_3 is considered fixed to the co-ordinate system. In other words, this boundary will move downwards with the same speed as the falling speed of the block if one observes the phenomenon in an inertial co-ordinate system.

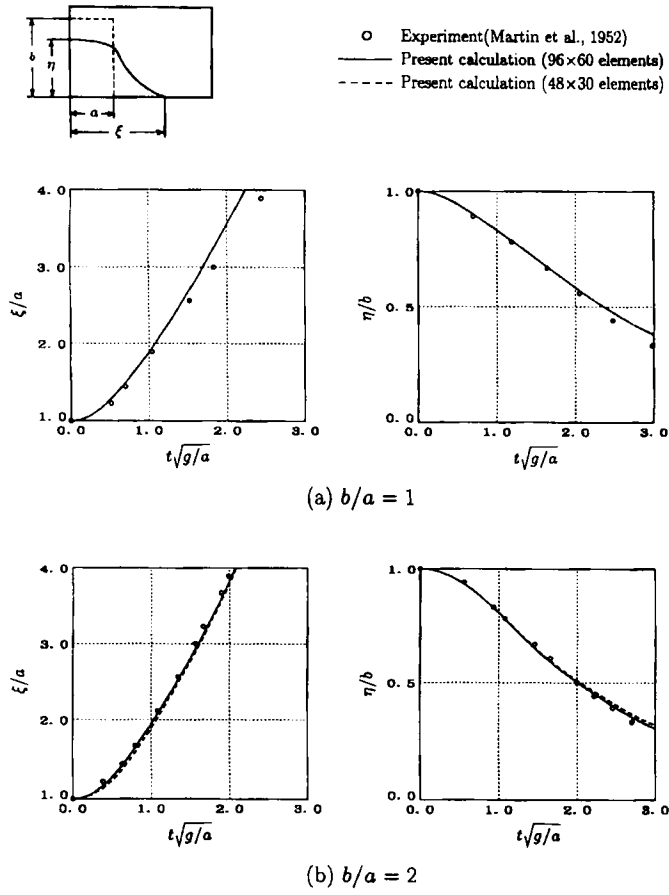


Figure 10. Comparison of computational and experimental time histories of water front location and water column height

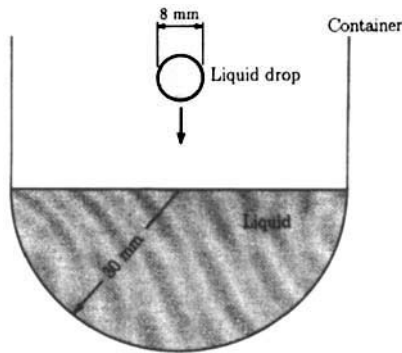


Figure 11. A liquid drop falling towards a still liquid surface in a container

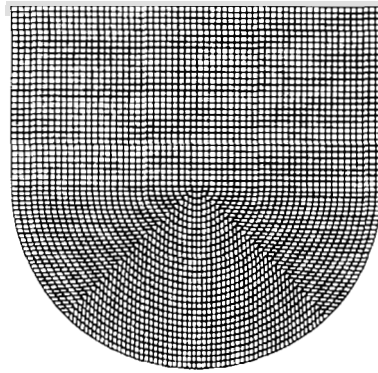


Figure 12. Subdivision of computational domain into four-node elements

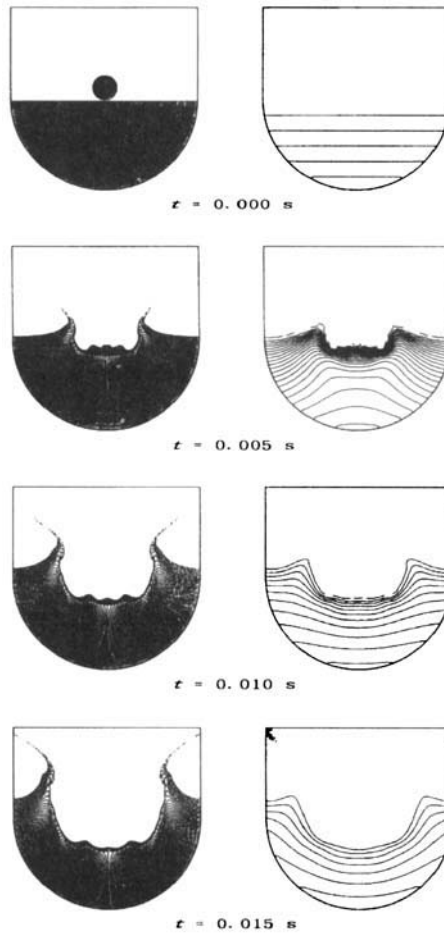


Figure 13. Marker distributions showing splash of a liquid drop (left) and corresponding pressure contours (right)

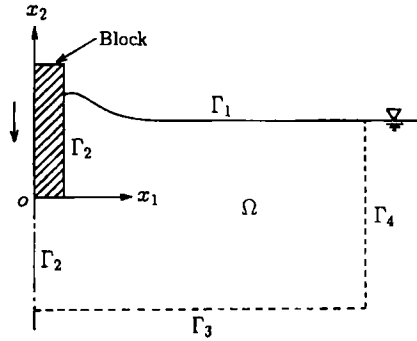


Figure 14. Computational model for water entry problem of a rigid block

Let u_i be the liquid velocity relative to the co-ordinate system. Then the momentum equation (1) is replaced by

$$\frac{\partial u_i}{\partial t} + u_j u_{i,j} = \sigma_{ij,j} + f_i + A_i \quad \text{in } \Omega, \quad (34)$$

where $A_i(t)$ denotes the x_i -component of acceleration of the block. The boundary condition on Γ_1 is given by equation (4) and the free slip condition (5) is imposed on Γ_2 and Γ_4 . The boundary condition on Γ_3 is expressed as

$$u_i = -V_i, \quad (35)$$

where $V_i(t)$ is the x_i -component of velocity of the block.

Initially the liquid velocity is set as

$$u_i(x_1, x_2, 0) = -V_i(0) \quad \text{in } \Omega \quad (36)$$

and the pressure in the liquid is given as

$$p(x_1, x_2, 0) : \text{hydrostatic pressure} \quad \text{in } \Omega. \quad (37)$$

Owing to the symmetry of the problem, the acceleration A_1 and the velocity V_1 are identically zero. The vertical acceleration A_2^{n+1} at time t^{n+1} is calculated by using the equation of motion of the block as

$$A_2^{n+1} = \frac{F_2^{n+1}}{M_B} - g, \quad (38)$$

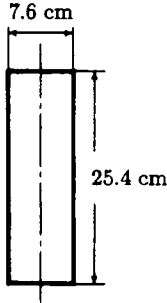
where M_B is the mass of the block, F_2 is the x_2 -component of pressure drag and g is the gravitational acceleration. The vertical velocity V_2 and the penetrating distance H of the block measured downwards from the still free surface are computed by the equations

$$V_2^{n+1} = V_2^n + \frac{\Delta t}{2} (A_2^{n+1} + A_2^n), \quad (39)$$

$$H^{n+1} = H^n - \frac{\Delta t}{2} (V_2^{n+1} + V_2^n). \quad (40)$$

The numerical calculation is carried out for two cases with different block masses and the results are compared with available experimental data. The block size and some computational conditions are summarized in Table I.

Table I. Computational conditions for water entry problem of a rigid block

Block size		
mass of block	102.2 g	153 g
Initial impact velocity	-33 cm s^{-1}	-30 cm s^{-1}
Size of initial liquid region	57 cm \times 23 cm	
Initial number of markers	15920 (18 per element)	
Time increments	$5 \times 10^{-3} \text{ s}$	$4 \times 10^{-3} \text{ s}$

The finite element mesh pattern is shown in Figure 15. Figure 16 shows the marker distributions and pressure contours at different time instants. The minimum pressure of the contour is 100 Pa and its increment is 100 Pa. The initial number of markers is given in Table I and additional markers are entered into the liquid region through the boundary Γ_3 according to the magnitude of the velocity V_2 every time step.

Figure 17 shows the time histories of three quantities: the penetration distance of the block, $H(t)$, the acceleration of the block, $A_2(t)$, and the pressure acting at the origin o . Open circles denote the present computed results and free curves represent the experimental data of Cheng and Leland.²⁰ The

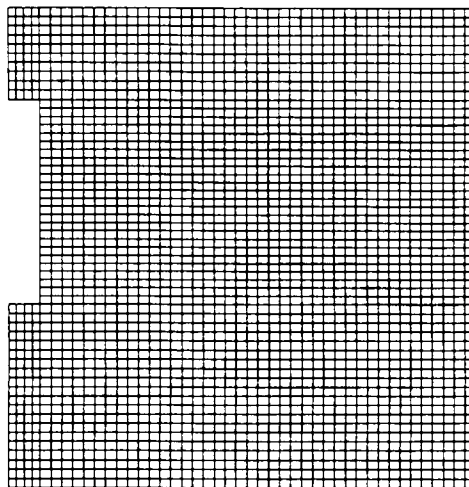


Figure 15. Finite element subdivision of computational domain

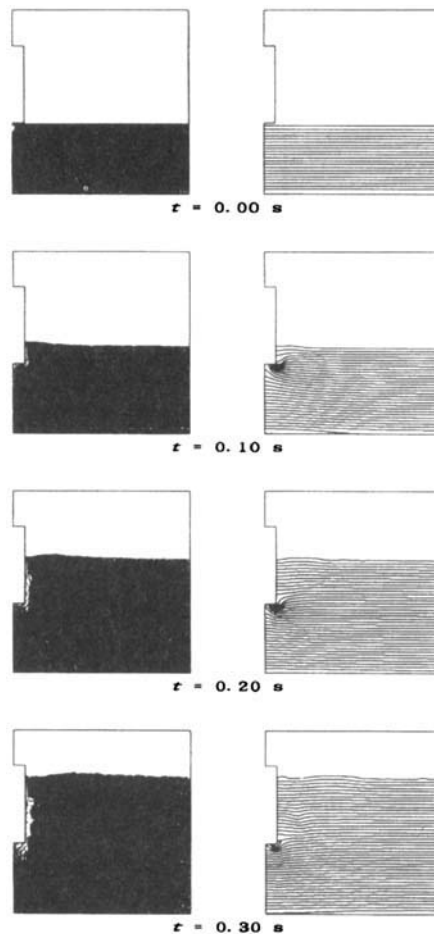


Figure 16. Marker distributions (left) and pressure contours (right) at different time instants (mass of block, 102.2 g)

agreement between the computational and experimental results is satisfactory in every comparison except for the peak value of pressure and its appearance time.

6. CONCLUDING REMARKS

The finite element method has been applied to the numerical simulation of the dynamic behaviour of a liquid with free surfaces. The proposed simulation method has two key features. One is the particle representation of liquid regions, which makes it easy to treat the arbitrary deformation of liquid regions numerically. The other is a new type of element-by-element algorithm for solving the flow equations in time. Owing to this algorithm, the assemblage of any global matrices can be avoided in the present finite element formulation and the required computer memory is reduced sharply. The numerical results have demonstrated the usefulness of the present method.

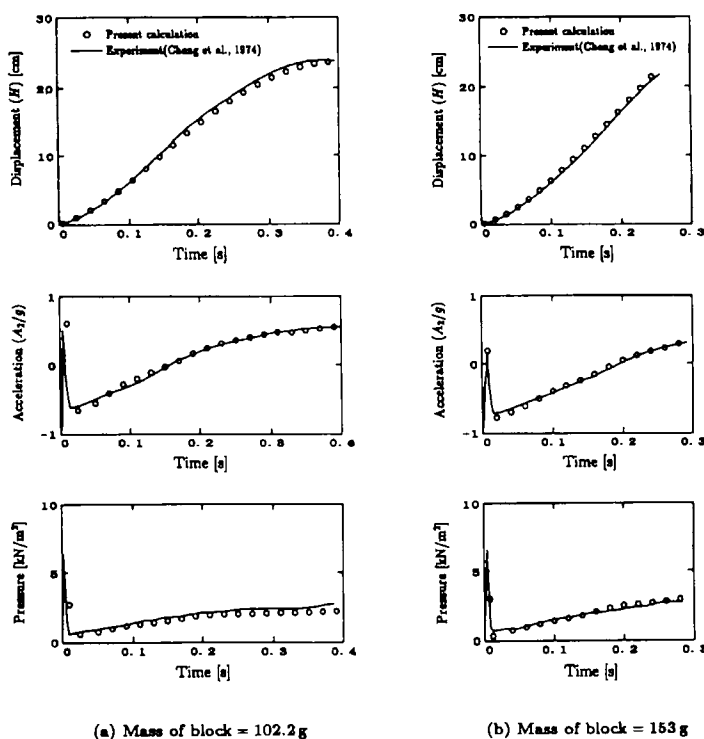


Figure 17. Time histories of displacement and acceleration of block and pressure on block

ACKNOWLEDGEMENTS

The authors would like to express their sincere gratitude to the reviewers for their valuable comments. The authors also wish to express their sincere thanks to Mr M. Ono for his help given to them during the preparation of the manuscript.

This research has been supported by the Japanese Ministry of Education, Science and Culture under a Grant-in-Aid for Scientific Research (C).

REFERENCES

1. R. K.-C. Chan, 'A generalized arbitrary Lagrangian-Eulerian method for incompressible flows with sharp interfaces', *J. Comput. Phys.*, **58**, 311-331 (1975).
2. J. Donea, S. Giuliani and J. P. Halleux, 'An arbitrary Lagrangian-Eulerian finite element method for transient dynamic fluid-structure interactions', *Comput. Methods Appl. Mech. Eng.*, **33**, 689-723 (1982).
3. W. K. Liu, H. Chang, J. S. Chen and T. Belytschko, 'Arbitrary Lagrangian-Eulerian Petrov-Galerkin finite elements for nonlinear continua', *Comput. Methods Appl. Mech. Eng.*, **68**, 259-310 (1988).
4. T. E. Tezduyar, M. Behr and J. Liou, 'A new strategy for finite element computations involving moving boundaries and interfaces—the deforming-spatial-domain/space-time procedure: I. The concept and the preliminary tests', *Comput. Methods Appl. Mech. Eng.*, **94**, 339-351 (1992).
5. T. E. Tezduyar, M. Behr, J. Liou and S. Mittal, 'A new strategy for finite element computations involving moving boundaries and interfaces—the deforming-spatial-domain/space-time procedure: II. Computation of free-surface flows, two-liquid flows, and flows with drifting cylinders', *Comput. Methods Appl. Mech. Eng.*, **94**, 353-371 (1992).
6. C. W. Hirt, B. D. Nichols and N. C. Romero, 'SOLA—a numerical solution algorithm for transient fluid flows', *Los Alamos Scientific Laboratory Rep. LA-5852*, University of California, 1975.
7. C. W. Hirt and B. D. Nichols, 'Volume of fluid (VOF) method for the dynamics of free boundaries', *J. Comput. Phys.*, **39**, 201-225 (1981).

8. J. E. Welch, F. H. Harlow, J. P. Shannon and B. J. Daly, 'The MAC method—a computing technique for solving viscous, incompressible, transient fluid-flow problems involving free surfaces', *Los Alamos Scientific Laboratory Rep. LA-3425*, University of California, 1966.
9. H. P. Wang and R. T. McLay, 'Automatic remeshing scheme for modeling hot forming process', *J. Fluid Eng., Trans. ASME*, **108**, 465–469 (1986).
10. F. Muttin, T. Coupez, M. Bellet and J. L. Chenot, 'Lagrangian finite-element analysis of time-dependent viscous free-surface flow using an automatic remeshing technique: application to metal casting flow', *Int. j. numer. methods eng.*, **36**, 2001–2015 (1993).
11. A. A. Amsden and F. H. Harlow, 'The SMAC method: a numerical technique for calculating incompressible fluid flows', *Los Alamos Scientific Laboratory Rep. LA-4370*, University of California, 1970.
12. R. K. -C. Chan and R. L. Street, 'A computer study of finite-amplitude water waves', *J. Comput. Phys.*, **6**, 68–94 (1970).
13. H. Miyata, 'Finite-difference simulation of breaking waves', *J. Comput. Phys.*, **65**, 179–214 (1986).
14. J. A. Viecegli, 'A computing method for incompressible flows bounded by moving walls', *J. Comput. Phys.*, **8**, 119–143 (1971).
15. A. Mizukami, 'Some integration formulas for a four-node isoparametric element', *Comput. Methods Appl. Mech. Eng.*, **59**, 111–121 (1986).
16. T. Nakayama and H. Yazaki, 'Finite element analysis of the transient motion of stratified viscous fluids under gravity', *Comput. Mech.*, **12**, 123–133 (1993).
17. A. J. Chorin, 'Numerical solution of the Navier–Stokes equations', *Math. Comput.*, **22**, 745–762 (1968).
18. B. D. Nichols and C. W. Hirt, 'Calculating three-dimensional free surface flows in the vicinity of submerged and exposed structures', *J. Comput. Phys.*, **12**, 234–246 (1973).
19. J. C. Martin and W. J. Moyce, 'An experimental study of the collapse of liquid columns on a horizontal plane', *Philos. Trans. R. Soc. Lond. A*, **244**, 312–324 (1952).
20. R. Y.-K. Cheng and T. J. W. Leland, 'Numerical solution for low-velocity penetration of rigid body into still fluid', in C. A. Brebbia and J. J. Connor (eds), *Numerical Methods in Fluid Dynamics*, Pentech, London, 1974, pp. 272–289.