INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2009; **60**:1032–1054
Published online 4 November 2008 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/fld.1937

Some improvements on free surface simulation by the particle finite element method

B. Tang, J. F. Li and T. S. Wang*, †

School of Aerospace, Tsinghua University, Beijing 100084, China

SUMMARY

The Lagrangian method has become increasingly popular in numerical simulation of free surface problems. In this paper, after a brief review of a recent Lagrangian method, namely the particle finite element method, some issues are discussed and some improvements are made. The least-square finite element method is adopted to simplify the solving of the Navier–Stokes equations. An adaptive time method is derived to obtain suitable time steps. A mass correction procedure is imported to improve the mass conservation in long time calculations and time discretization scheme is adopted to decrease the pressure oscillations during the calculations.

Finally, the method is used to simulate a series of examples and the results are compared with the commercial FLOW3D code. Copyright © 2008 John Wiley & Sons, Ltd.

Received 1 January 2008; Revised 20 August 2008; Accepted 27 August 2008

KEY WORDS: free surface; particle finite element method; adaptive time method; mass correction; pressure oscillations; least-square finite element method

1. INTRODUCTION

Free surface problems are of great importance in various engineering applications, including sloshing in liquid containers, spillways in dams, ship hydrodynamics, mold filling processes and many other domains.

The numerical simulation of these phenomena can be accomplished by solving the Navier–Stokes (N–S) equations with interfaces. On one hand the numerical methods in the case of long time calculations of N–S equations are still a considerable task. On the other hand, there are specific problems due to the presence of an interface.

*Correspondence to: T. S. Wang, School of Aerospace, Tsinghua University, Beijing 100084, China.
†E-mail: tswang@tsinghua.edu.cn

Contract/grant sponsor: Publishing Arts Research Council; contract/grant number: 98-1846389
Contract/grant sponsor: National Natural Science Foundation; contract/grant numbers: 10302013, 10572022

There are two kinds of methods for interface simulation: fixed or moving grid methods [1]. In fixed grid methods, there is a predefined grid that does not move with the interface. The interface has to somehow cut across this fixed grid. In addition to solving the N–S equations, an additional method must be imposed to capture the interface. Most of the literature is concerned with volume-of-fluid (VOF) [2] methods or level-set [3] methods, and many commercial codes such as FLOW3D and Fluent use these methods to represent interfaces. Another kind of technique, known as moving grid or tracking methods, relies on the use of markers, whose positions are updated with the current computed fluid velocity field. This simplifies the analysis near the interface. The problem of moving grid methods is that when the interface undergoes large deformations, the grid has to be remeshed. However, moving grid techniques have been presented recently because of their simplicity, including the meshless method [4, 5], the natural element method (NEM) [6] and the particle finite element method (PFEM) [7].

In this paper, PFEM is described and some improvements in the adaptive time step, mass conservation and method for decreasing the pressure oscillations are introduced. In Section 2, PFEM is outlined briefly. In Section 3.1, the least-square finite element method (LSFEM) is used to solve the N–S equations instead of the finite calculus (FIC) method proposed by PFEM. This substitution is not necessary, but can reduce the difficulty of the programming considerably. In Section 3.2, a new kind of adaptive time-step scheme is derived to improve the robustness of PFEM and the efficiency of the calculation. In Section 3.3, a mesh-pull method is imported to ensure the mass conservation. In Section 3.4, a new method is proposed to avoid the pressure oscillation due to the rough boundary conditions. The whole method is summarized in Section 4, and numerical examples are given in Section 5.

2. THE PFEM

One feature of the PFEM is that the fluid equations are solved by the FIC method, which is a stabilized fractional step method [7]. Moreover, it uses an innovative extended Denaulay tessellation (EDT) algorithm to generate the meshes, and applies the natural neighbor method in the integration of the elements. Then the α-shape method is used to obtain the shape of the set of nodes.

2.1. The FIC method

The Navier–Stokes equations can be written as

$$\nabla \cdot \mathbf{v} = 0 \tag{1}$$

$$\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t} - \nu \nabla^2 \mathbf{v} + \frac{1}{\rho} \nabla p = \mathbf{b} \tag{2}$$

where $\mathbf{v}$ is the velocity field, p is the pressure, ρ is the density of the fluid, $\mathbf{b}$ is the body force, ν is the kinematical viscosity coefficients and t represents time.

The FIC method is based on the modification of the above equations in a finite region, and solves them by a stabilized fractional step method. The details of this method are described in [7, 8].

However, the modified equations are not easy to solve because the stabilization terms are difficult to calculate [9, 10].

Alternatively, the LSFEM is another numerical method different from the Galerkin method. It can be used to get stabilized solutions without any of the stabilization terms as FIC. This method

DOI: 10.1002/fld

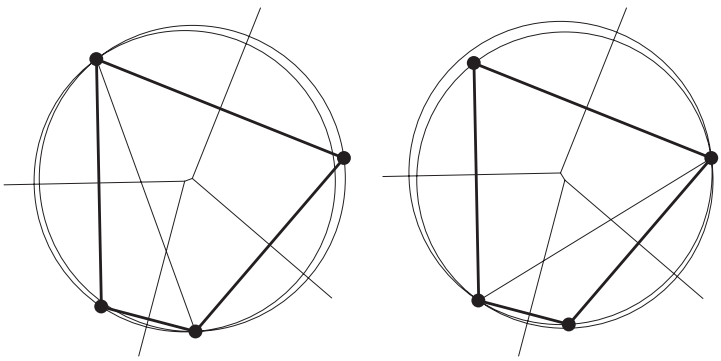

Figure 1. The EDT algorithm combines two close triangles and generates a unique mesh.

is adopted in the present work to simplify the coding of PFEM [11]. The details of LSFEM are described in Section 3.1.

2.2. The EDT

The details of this method are reported in [12]. It generates the triangular mesh on the basis of the standard Denaulay tessellation (DT) [13]. A serious problem of the DT is that the tessellation may not be unique and may result in near zero volumes. The EDT combines two triangles if they are 'close' to each other. The criterion of 'close' is that the distance between the circumcircle centers of these two triangles is not larger than a small constant multiplied by their average radius. As illustrated in Figure 1, if the two triangles are close enough, they are combined to form a quadrangle, and this quadrangle can be merged to form a polygon too. The numerical test [12] shows that the EDT: (a) is unique for a set of node distributions; (b) is formed by a polyhedron with nonzero volume and (c) is obtained in a bounded time of order n.

2.3. Interpolation shape functions

Since the elements generated by EDT are not standard triangles, sometimes the non-Sibson interpolation is used [14, 15]. This interpolation function is derived from the NEM but has a small difference: The values at the interpolation point are interpolated from all nodes 'near' [14] this point by NEM, and the regions derived from these 'near' nodes may be overlapped or gaped. But in the PFEM, the values at the interpolation point are interpolated from the element that surrounds this point. Thus the standard Gaussian integration can be taken in each element for PFEM, while the integration in NEM is somewhat difficult [16]. Figure 2 illustrates the definition of the functions as

$$\phi_p \mathbf{x} = s_p / \|\mathbf{n}_p - \mathbf{x}\| = s_p / h_p \tag{3}$$

and the shape function is

$$N_p = \phi_p \Big/ \sum_q \phi_q \tag{4}$$

It can be easily proven that this shape function satisfies the partition of unity property and has linear completeness, and the derivatives of the function can be calculated by a geometrical method [15].

DOI: 10.1002/fld

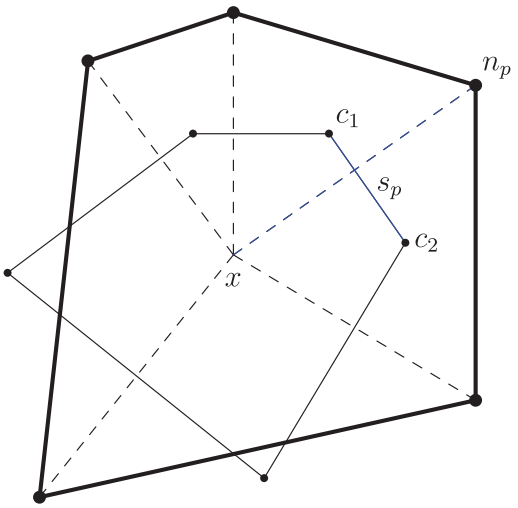

Figure 2. Schematic figure of the non-Sibson interpolation.

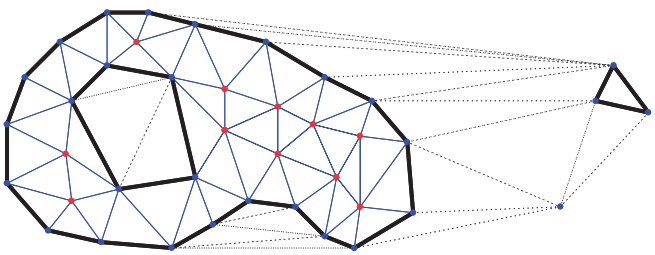

Figure 3. Recognition of boundary surfaces with individual particles by α-shape method.

2.4. The α-shape boundary

Recently, the α-shape method has been used intensively, such as the NEM [6], because it can recognize the shape of the free surface automatically and the essential boundary conditions can be imposed. As described in [17], this method is used by PFEM in the following way:

> *All elements that are on a sphere* (*circle*) *with a radius* $r(e)$ *greater than* $\alpha h(e)$, *are considered as inactive elements*.

In this criterion, α is a parameter close to one, typically $\alpha = 1.2$. $r(e)$ is the radius of the circumcircle of the element and $h(e)$ is the mean value of the characteristic length of the particles that are on this element.

Once the decision of which of the elements is inactive has been made, the boundary nodes must be defined. Here

> *A boundary edge is defined if only one activate element contains this edge. All nodes on these boundary edges are boundary nodes*.

DOI: 10.1002/fld

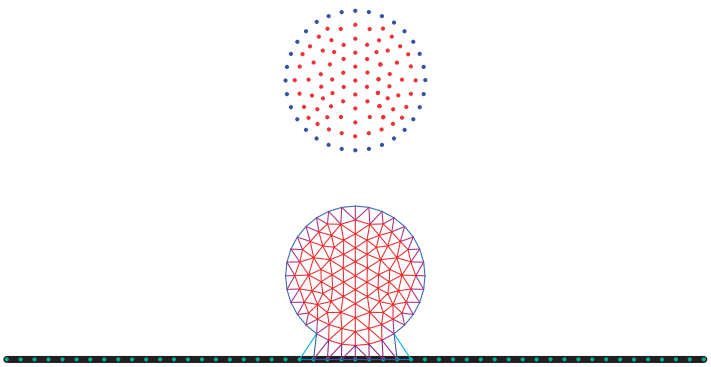

Figure 4. The α-shape technique used for impact recognition.

A feature of the α-shape technique is that it can not only recognize the free surface but can also identify isolated fluid particles outside the main fluid domain by scanning all the active elements. Figure 3 shows a schematic example of the boundary surface from a given collection of nodes. Here all lines are the edges connected by EDT. After the implementation of the α-shape method, the solid lines represent the edges of the active elements, and the dotted lines represent the edges of the inactive elements. EDT always gives the convex hull of the nodes (all lines), but the boundary recognition algorithm captures the external free surface as well as the internal boundary surface (solid lines), and it also captures the individual nodes (single node or triangle), which depart from the main fluid domain.

2.5. Treatment of contact between fluid and solid boundaries

Another important feature of the α-shape technique is related to the contact problem. The contact can be solved naturally as illustrated in Figure 4. At the beginning the fluid (dotted circle at the top) is above the solid (solid line). When the fluid drops and is going to impact the solid (lined circle at the bottom), a few new elements between the fluid and the solid appears and can naturally simulate the impact.

In PFEM, sometimes it is necessary to add or delete some nodes. This is called 'Joining and breaking particles' in the literature [15]. The following criterion has been used in every time step as follows:

Delete nodes: The node is not added in EDT if there is already a previous point at a distance $d<\lambda h(\mathbf{x})$, in which $\lambda=0.5$ is a constant parameter and $\mathbf{x}$ is the position of this node.
Add nodes: If there is an element whose radius $r(e)>Ch(e)$, a point is added to its center and assigned with the values interpolated from the element, in which $C=0.8$ is a constant parameter.

The procedure in adding nodes are similar to the step in identifying inactive elements. The difference is the coefficients of these two procedures. When $r(e)>1.2h(e)$, the element is seemed as inactive, while when $1.2h(e)\geqslant r(e)>0.8h(e)$, an addition node will be added into this element.

With the above techniques, Onate and Idelsohn obtained many exciting results (http://www.cimne.com/pfem/).

DOI: 10.1002/fld

3. THE IMPROVEMENTS OF PFEM

Although the results obtained by PFEM are very exciting, some improvements can still be made, such as the adaptive time step, the mass conservation and the pressure correction. Futhermore, the FIC can be replaced by the LSFEM because of simplicity.

3.1. The LSFEM

This procedure is not necessary but it can greatly simplify the coding of PFEM. The stabilized fractional step method used in PFEM is very complex because choosing the stabilization terms is difficult [9, 10].

To get the C_0 interpolation function, Equations (1) and (2) are converted to one-order equations by imposing an assistant variable $\boldsymbol{\omega}=\nabla\times\mathbf{v}$, and then temporally discreting by finite difference method. We get

$$\begin{aligned}\nabla\cdot\mathbf{v}^{n+1}&=0\\ \frac{\mathbf{v}^{n+1}-\mathbf{v}^n}{\Delta t}+\theta\left(\frac{1}{\rho}\nabla p^{n+1}+v\nabla\times\boldsymbol{\omega}^{n+1}\right)+(1-\theta)\left(\frac{1}{\rho}\nabla p^n+v\nabla\times\boldsymbol{\omega}^n\right)&=\mathbf{b}^{n+1}\\ \nabla\cdot\boldsymbol{\omega}^{n+1}&=0\end{aligned}\tag{5}$$

where $\theta=\frac{1}{2}$ is the Crank–Nicolson (CN) formulation and $\theta=1$ is the backward Euler (BE) method.

To facilitate subsequent analysis, Equation (5) can be written in the matrix form

$$\mathscr{L}\mathbf{u}=\mathbf{f}\tag{6}$$

where $\mathscr{L}$ is a first-order partial differential operator

$$\mathscr{L}=\sum_{i=1}^{n_d}\mathscr{L}_i\frac{\partial\mathbf{u}}{\partial x_i}+\mathscr{L}_0\mathbf{u}\tag{7}$$

here n_d is the dimension of the space. In two-dimensional cases, the $\mathscr{L}$ of N–S equations can be written as

$$\mathbf{u}=\begin{pmatrix}u\\ v\\ p\\ \omega\end{pmatrix},\quad \mathscr{L}_0=\begin{pmatrix}0&0&0&0\\ \frac{1}{\Delta t}&0&0&0\\ 0&\frac{1}{\Delta t}&0&0\\ 0&0&0&1\end{pmatrix},\quad \mathscr{L}_1=\begin{pmatrix}1&0&0&0\\ 0&0&\frac{\theta}{\rho}&0\\ 0&0&0&-\theta v\\ 0&-1&0&0\end{pmatrix}$$

$$\mathscr{L}_2=\begin{pmatrix}0&1&0&0\\ 0&0&0&\theta v\\ 0&0&\frac{\theta}{\rho}&0\\ 1&0&0&0\end{pmatrix},\quad \mathbf{f}=\begin{pmatrix}0\\ b_x^{n+1}\\ b_y^{n+1}\\ 0\end{pmatrix}+\begin{pmatrix}0\\ -(1-\theta)\left(\frac{\partial p^n}{\partial x}+v\frac{\partial\omega^n}{\partial y}\right)\\ -(1-\theta)\left(\frac{\partial p^n}{\partial y}-v\frac{\partial\omega^n}{\partial x}\right)\\ 0\end{pmatrix}=\mathbf{f}_1+\mathbf{f}_2$$

DOI: 10.1002/fld

As a result of the Lagrangian description, no nonlinear convection terms are needed and $\mathscr{L}$ and $\mathbf{f}$ are constants matrix at each time step, eliminating the procedure for linearization.

They are first-order linear partial differential equations and can be discrete by the finite element method. For a given interpolation function φ_i, the variable values $\mathbf{u}_\mathrm{e}(\mathbf{x})$ in each element can be expressed as

$$\mathbf{u}_\mathrm{e}(\mathbf{x}) = \sum_{j=1}^{N_n} \varphi_j(\mathbf{x})\,\mathbf{u}_j^\mathrm{e} \tag{8}$$

where N_n is the number of elements and $\mathbf{u}_j^\mathrm{e}$ are the four variables at jth nodes of each element. Therefore, Equation (6) in each element can be written as

$$\begin{aligned}\mathscr{L}_\mathrm{e}\mathbf{u}_\mathrm{e} &= \mathscr{L}_\mathrm{e}\left(\sum_{j=1}^{N_n}\varphi_j(\mathbf{x})\mathbf{u}_j^\mathrm{e}\right) = \sum_{j=1}^{N_n}\mathscr{L}_0\varphi_j(\mathbf{x})\mathbf{u}_j^\mathrm{e} + \sum_{j=1}^{N_n}\sum_{k=1}^{n_d}\mathscr{L}_k\frac{\partial\varphi_j(\mathbf{x})}{\partial x_k}\mathbf{u}_j^\mathrm{e}\\ &= \sum_{j=1}^{N_n}\mathbf{B}_j\mathbf{u}_j^\mathrm{e} = \mathbf{f}_\mathrm{e}\end{aligned} \tag{9}$$

where

$$\mathbf{B}_j = \mathscr{L}_0\varphi_j + \mathscr{L}_1\frac{\partial\varphi_j}{\partial x} + \mathscr{L}_2\frac{\partial\varphi_j}{\partial y}$$

The finite element solution $\mathbf{u}_\mathrm{e}$ on the nodes cannot satisfy Equation (9) accurately, and it will have a residual

$$\mathbf{R}_\mathrm{e} = \sum_{j=1}^{N_n}\mathbf{B}\mathbf{u}_j^\mathrm{e} - \mathbf{f}_\mathrm{e} \tag{10}$$

The meaning of LSFEM is to minimize the Equation (9) in the sense of least square, i.e. to minimize the functional

$$I(\mathbf{u}_\mathrm{e}) = \int_\mathrm{e}\left(\sum_{j=1}^{N_n}\mathbf{B}_j\mathbf{u}_j^\mathrm{e} - \mathbf{f}_\mathrm{e}\right)^\mathrm{T}\left(\sum_{j=1}^{N_n}\mathbf{B}_j\mathbf{u}_j^\mathrm{e} - \mathbf{f}_\mathrm{e}\right)\mathrm{d}\Omega \tag{11}$$

To get the necessary condition, it must have

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}\mathbf{u}_i^\mathrm{e}}I(\mathbf{u}_\mathrm{e}) &= \int_\mathrm{e}\frac{\mathrm{d}}{\mathrm{d}\mathbf{u}_i^\mathrm{e}}\left(\left(\sum_{j=1}^{N_n}\mathbf{B}_j\mathbf{u}_j^\mathrm{e} - \mathbf{f}_\mathrm{e}\right)^\mathrm{T}\left(\sum_{j=1}^{N_n}\mathbf{B}_j\mathbf{u}_j^\mathrm{e} - \mathbf{f}_\mathrm{e}\right)\right)\mathrm{d}\Omega\\ &= 2\int_\mathrm{e}\left(\frac{\mathrm{d}}{\mathrm{d}\mathbf{u}_i^\mathrm{e}}\left(\sum_{j=1}^{N_n}\mathbf{B}_j\mathbf{u}_j^\mathrm{e} - \mathbf{f}_\mathrm{e}\right)^\mathrm{T}\right)\left(\sum_{j=1}^{N_n}\mathbf{B}_j\mathbf{u}_j^\mathrm{e} - \mathbf{f}_\mathrm{e}\right)\mathrm{d}\Omega\\ &= 2\int_\mathrm{e}\mathbf{B}_i^\mathrm{T}\left(\sum_{j=1}^{N_n}\mathbf{B}_j\mathbf{u}_j^\mathrm{e} - \mathbf{f}_\mathrm{e}\right)\mathrm{d}\Omega = 0\end{aligned} \tag{12}$$

DOI: 10.1002/fld

We write the sum terms in Equation (12) as

$$\sum_{j=1}^{N_n} \mathbf{B}_j \mathbf{u}_j^{\mathrm{e}} = (\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_{N_n})(\mathbf{u}_1^{\mathrm{e}}, \mathbf{u}_2^{\mathrm{e}}, \ldots, \mathbf{u}_{N_n}^{\mathrm{e}})^{\mathrm{T}} = (\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_{N_n})\mathbf{U}_{\mathrm{e}} \tag{13}$$

Then Equation (12) can be written in the following algebraic matrix form:

$$\mathbf{K}_{\mathrm{e}}\mathbf{U}_{\mathrm{e}} = \mathbf{F}_{\mathrm{e}} \tag{14}$$

The corresponding element stiffness matrix and element load vector are

$$\mathbf{K}_{\mathrm{e}} = \int_{\mathrm{e}} (\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_{N_n})^{\mathrm{T}} (\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_{N_n})\, \mathrm{d}\Omega \tag{15}$$

$$\mathbf{F}_{\mathrm{e}} = \int_{\mathrm{e}} (\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_{N_n})^{\mathrm{T}} \mathbf{f}_{\mathrm{e}}\, \mathrm{d}\Omega \tag{16}$$

The global stiffness matrix and load vector are obtained after finite element integration and the final algebraic equation is

$$\mathbf{K}\mathbf{U} = \mathbf{F} \tag{17}$$

The global stiffness matrix is symmetric positive definite and can be effectively solved by the preconditioned conjugate gradient method [11, 18].

Sometimes addition coefficient is added to Equation (1) to enhance the importance of the continuity equation [19, 20]. This method is called the weighted LSFEM and can be used to improve the mass conservation notable. The coefficient is chosen as 10 in this paper.

3.2. The adaptive time step

It was presented in PFEM [7] that the time increment size was chosen as

$$\Delta t = \min \Delta t_i \quad \text{with } \Delta t_i = \frac{h_i^{\min}}{|\mathbf{v}|} \tag{18}$$

where $h_i^{\min}$ is the distance between node i and the closest one in the mesh, and $\mathbf{v}$ is the velocity of node i.

It was declared that 'in some problems it was useful to define a layer of nodes adjacent to the external boundary in the fluid where the condition of prescribed velocity was imposed' [7]. This layer of nodes seems somewhat strange because the rigid nodes can naturally prevent the penetration of the water nodes into the solid boundaries very well and the layer of nodes is redundant if a suitable time step is used.

In order to obtain a suitable time-step size in each step, three kinds of adaptive time algorithms are considered in this paper.

1. Assume that the jth node has a velocity $\mathbf{v}_j$ and h_j is the distance from the nearest node to itself. Let $\Delta t < \min_j(|v_j|/h_j)$, for example, $\Delta t = \frac{1}{2}\min(|v_j|/h_j)$. This method, which is the method used by PFEM, can be imposed easily, but Δt is too small in some cases such as rigid body movement. Moreover, the fluid may penetrate the solid surface as illustrated in Figure 5, where the points at $\mathbf{x}_1$ and $\mathbf{x}_2$ denote the solid boundary. Fluid particle at $\mathbf{x}$ arrives at the new position $\mathbf{x}'$, where the fluid particle has crossed through the solid boundary.

DOI: 10.1002/fld

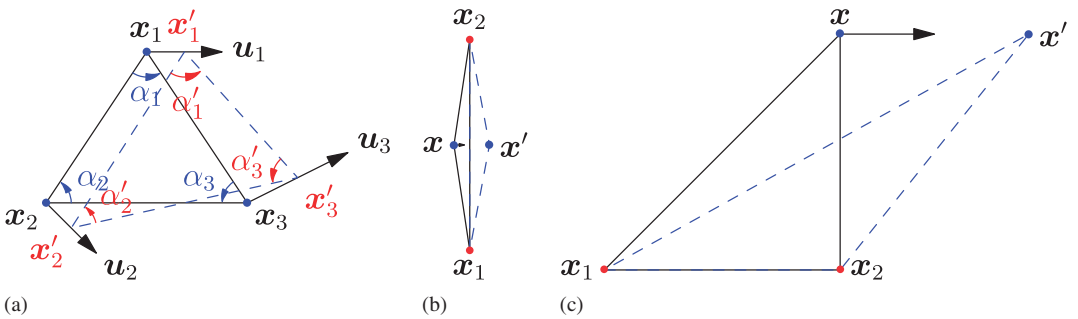

Figure 5. Simple mesh for explaining adaptive time algorithm.

2. Assume that the time step is Δt. Then all nodes on the element illustrated in Figure 5(a) have new positions $\mathbf{x}'=\mathbf{x}+\mathbf{v}\Delta t$, respectively. The time step should guarantee that the element after deformation is not singular. So the critical condition is that the area of the element after deformation is zero; i.e.

$$\begin{vmatrix} 1 & x_1+u_1\Delta t & y_1+v_1\Delta t \\ 1 & x_2+u_2\Delta t & y_2+v_2\Delta t \\ 1 & x_3+u_3\Delta t & y_3+v_3\Delta t \end{vmatrix} = a\Delta t^2+b\Delta t+c=0 \tag{19}$$

where

$$a=u_2v_3-v_2u_3-u_1v_3+v_1u_3+u_1v_2-v_1u_2$$

$$b=x_2v_3+u_2y_3-y_2u_3-v_2x_3-x_1v_3-u_1y_3$$
$$+y_1u_3+v_1x_3+x_1v_2+u_1y_2-y_1u_2-v_1x_2$$

$$c=x_2y_3-y_2x_3-x_1y_3+y_1x_3+x_1y_2-y_1x_2$$

Then Δt can be solved from this quadratic equation. If the solution is negative or imaginary, there is no limit to the time step of this element, while if the solution is positive, $\frac{1}{2}$ of this value is chosen as the time step.

This method can prevent penetration and is useful in most cases, except that the time step obtained is huge in situations such as that in Figure 5(c).

3. As illustrated in Figure 5(a), assume that all the angles after deformation are larger than half of their initial values, respectively. Take α_1 as an example; its original value is $\alpha_1=\text{atan2}(|\mathbf{r}_{21}\times\mathbf{r}_{31}|,\mathbf{r}_{21}\cdot\mathbf{r}_{31})$, and the angle after deformation is set to $\alpha_1'=\frac{1}{2}\alpha$. An equation like Equation (19) can be derived and solved to get a suitable time step.

In this paper algorithms (2) and (3) are utilized together; algorithm (2) is used for inactive elements to prevent the penetration and to catch the impact, and algorithm (3) for active elements to obtain the time step. The time step obtained in this way is large enough to improve the computation efficiency and small enough to prevent the penetration of the water nodes into the solid boundaries.

DOI: 10.1002/fld

This simple method to represent the water–wall contact is an attractive feature of the PFEM formulation. In addition, this method can be extended to three dimensions, by substituting a cubic equation for the quadratic equation (19).

3.3. The mass conservation

The PFEM does not conserve the volume exactly [21]. After updating the nodes, minute errors in the volume of the fluid may occur. These errors are very small in each time step but will accumulate after long time calculations. The error correction is based on shifting the recognized boundary surface (except the isolated nodes) in normal directions so that the volume at the current time is the same as that at the initial time [22]. Consequently, the coordinate at each free surface node is determined as follows:

$$\mathbf{x}_k = \mathbf{x} + k \cdot 10^{-4} h \mathbf{n}_x \quad \text{on } \Gamma_p \tag{20}$$

where $\mathbf{n}_x$ is the normal vector of the surface, h is the characteristic length of this node and k is the tuning number varying from -100 to 100. k is calculated at each time step to minimize

$$|m_k - m_{\text{origin}}| \tag{21}$$

where m_k is the mass with the tuning parameter k and m_{origin} is the initial state of the fluid mass.

The correction is acceptable in numerical calculation, though it is nonphysical and the maximum correction is 10^{-2} h at each time step. With this small correction, the relative mass error is improved to be $10^{-4} - 10^{-5}$ during the whole computation.

3.4. Decrease the pressure oscillations

The N–S equations are partial differential equations in both time and space. The discretization of space can be obtained by the finite element method, while the time discretization is commonly achieved through one of the methods from the treatment of ordinary differential equations, such as the BE and the CN scheme.

As seen from Equation (5) the parameter θ has to be chosen depending on the time-step scheme, e.g. $\theta = 1$ for the BE, and $\theta = \frac{1}{2}$ for the CN scheme. These two methods belong to the group of *one-step-θ-schemes*. The CN scheme is second order and is preferable, but it occasionally suffers from numerical oscillation because of its weak damping property, while the BE scheme is strongly damping but it is only first order.

For PFEM, the free surface is constructed for every time step, and the boundaries are sometimes rough. Calculation indicates that the rough boundary causes oscillation of the pressure. In fact, this phenomenon occurs not only in PFEM but also on other surface reconstruction algorithms and meshless methods. PFEM uses the BE method, as cited in [23]. As we all know, the BE is strongly numerically dissipated, which can be observed from the numerical examples below.

The multi-step method for PFEM is hard to construct because the mesh is different at each time step. Moreover, the nodes are added or deleted frequently. To overcome the oscillation of the CN and the dissipation of the BE, a new one-step method is used here.

First: The CN method is used to get the convergent $\mathbf{v}_{\text{CN}}$ and p_{CN} fields. The iteration procedure, which is used to get the convergent solution, is described in Section 4.
Second: The BE method is calculated again to get the convergent $\mathbf{v}_{\text{BE}}$ and p_{BE} fields.
Third: The final veocity and pressure results are $\mathbf{v}_{\text{CN}}$ and p_{BE}.

DOI: 10.1002/fld

Numerical tests indicate that if we take the $\mathbf{v}_{\mathrm{CN}}$ and p_{CN} fields as the initial solution in the *second* step, the convergence can be achieved in only one or two iterations, so we can simply drop the iteration in the *second* step.

The correction used here is simple and does not affect the efficiency very much as the *second* step runs only once, while the *first* step runs 3–4 times.

Unfortunately, the theoretical derivation of this method is lacking here. But numerical tests indicate that the $\mathbf{v}_{\mathrm{BE}}$ fields have a relative error of about 10^{-3} from the previous $\mathbf{v}_{\mathrm{CN}}$. And the comparison with commercial FLOW3D code indicates that this method is not only acceptable but also performs well in numerical calculations.

4. THE PROCEDURES

From all the methods specified above, the whole procedure in this paper is summarized in this section.

Although the nonlinear convective terms vanish in the Lagrangian description, the nonlinear effect does not disappear. It turns into another nonlinearity in the form of the movement of the finite element mesh. Therefore, the following iteration procedure must be carried out for each time step:

Step 1: Derive $(\mathbf{u}, p, \omega)^{n+1}$ from Equations (15)–(17) as

$$B^{n+1,j}_{\alpha\beta} B^{n+1,j}_{\beta\gamma} U^{n+1,j+1}_{\gamma i} = B^{n}_{\alpha\beta} f^{n}_{1\beta} + B^{n+1,j}_{\alpha\beta} f^{n+1,j}_{2\beta} \tag{22}$$

where

$$\mathbf{B}_j = \int_V \varphi_j \mathscr{L}_0 + \frac{\partial \varphi_j}{\partial x} \mathscr{L}_1 + \frac{\partial \varphi_j}{\partial y} \mathscr{L}_2 \, \mathrm{d}V$$

$$\mathbf{B} = (\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_{N_n})$$

$$\mathbf{f}_1 = \begin{pmatrix} 0 \\ \dfrac{u}{\Delta t} - (1-\theta)\left(\dfrac{\partial p}{\partial x} + v\dfrac{\partial \omega}{\partial y}\right) \\ \dfrac{v}{\Delta t} - (1-\theta)\left(\dfrac{\partial p}{\partial y} - v\dfrac{\partial \omega}{\partial x}\right) \\ 0 \end{pmatrix}, \quad \mathbf{f}_2 = \begin{pmatrix} 0 \\ f_x \\ f_y \\ 0 \end{pmatrix}$$

Note that the interpolation functions used for $n+1$ and n time steps are different at opposite sides of the equation. Here $(n+1, j)$ denotes the jth iteration at the $n+1$ time step, and the $(n+1, 0)$ step equals the nth step. The positions of the nodes are updated at the jth step, thus the interpolation functions on the left side of Equation (17) are also updated.

Step 2: Update the mesh nodes in a Lagrangian manner. From the definition of the velocity $\mathbf{v} = \mathrm{d}\mathbf{x}/\mathrm{d}t$, the new positions are derived from

$$\mathbf{x}_i^{n+1,j} = \mathbf{x}_i^n + 1/2(\mathbf{v}_i^n + \mathbf{v}_i^{n+1,j})\Delta t \tag{23}$$

DOI: 10.1002/fld

Step 3: Generate a new mesh and recognize the boundary by PFEM as described in Section 2.
Step 4: Check the convergence of the velocity by

$$\frac{|\mathbf{v}^{j+1}-\mathbf{v}^{j}|}{1+|\mathbf{v}^{j+1}|}<\varepsilon \tag{24}$$

where ε is a small error tolerance such as 10^{-6}. If this condition is satisfied, we move to *Step 5*, otherwise return to *Step 1* for the next iteration with $j \to j+1$.

Step 5: Correct the pressure and mass as described in Sections 3.4 and 3.3 and go to the next time step.

5. NUMERICAL EXAMPLES

5.1. Some notes on the numerical examples

There are three numerical examples proposed here. Namely, a circular drop impacts the surface of water, sloshing in a tank and the breaking of a dam with an obstacle. The results here are compared with the commercial FLOW3D code. In the setup of FLOW3D options, there are three types of algorithms in 'viscous stress evaluation': the explicit, the Jacobi implicit and the ADI algorithms. We found that the three algorithms have almost the same results. The Jacobi implicit algorithm was chosen here. In the calculation of 'momentum advection', FLOW3D has three types of algorithms: first order, second order and monotonicity preserving second-order algorithms. Numerical tests indicate that the latter two methods have some numerical instabilities. Thus, we used the first-order algorithm and in the 'pressure iterations', the SOR was used.

In the numerical calculation in PFEM, we set the range of the adaptive time step, i.e. 0.1 s as the maximum time step and 1e−4 s as the minimum time step.

In all the numerical examples the kinematic viscous coefficient was supposed to be $\mu=10^{-6}\,\mathrm{m}^2/\mathrm{s}$ and the gravity was $g=-9.8\,\mathrm{m/s}^2$.

5.2. Circular drop impacting the water

The method was verified by an example of a water drop with radius of 0.15 m (Figure 6). The water fell from a height of 0.7 m and impacted the free surface of water in a pool with dimensions of 2 m × 1 m. The results were compared with commercial FLOW3D code. The adaptive time-step method, the mass conservation properties and the pressure fields were examined.

5.2.1. The adaptive time-step method. The time step chosen in PFEM literature [10], the fixed time step and the adaptive time-step method are compared.

Figure 7 illustrates the time step in the original PFEM method and the method described in Section 3.2. The average time step is 0.0023 s in the original PFEM method, whereas it is 0.01265 s in the method proposed here. Figures 8 and 9 illustrate the mesh when a circular drop impacts the water. Figure 8 is with a fixed time step 0.01 s. The mesh was very poor near impact line but even with this mesh, the calculation could go on because of the continuous updating of the mesh and the procedure of 'adding' and 'deleting' points. However, near the boundary of the tank, the node would penetrate the rigid tank. To avoid this problem, a smaller time step must be chosen and no doubt it will decrease the computational efficiency.

DOI: 10.1002/fld

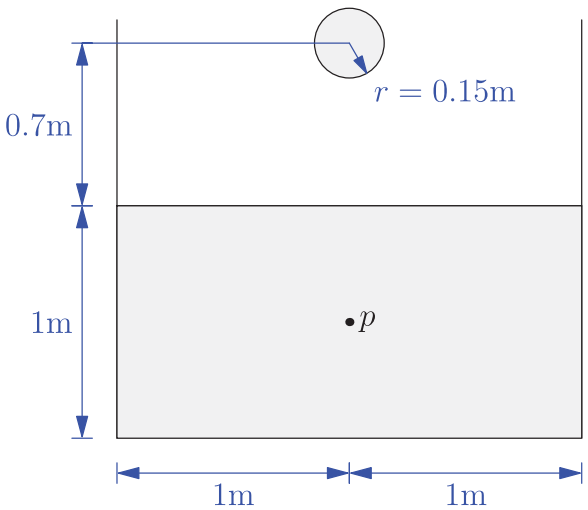

Figure 6. The sketch of a circular drop impacting the water.

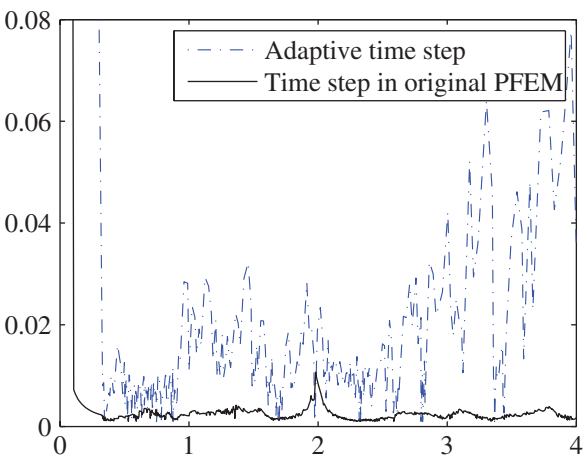

Figure 7. The time step used in the original PFEM.

The penetration does not appear in this example, but we direct the comparison in the dam break example shown in Figure 21.

5.2.2. The mass/volume conservation. The PFEM does not conserve the volume exactly [21]. The error of volume can be understood according to two aspects. The first is that extra unphysical mesh is generated by the α-shape method as shown in Figure 9. This is the original drawback of the α-shape method. This error can only be improved by the refinement of the mesh but cannot be completely eliminated. Fortunately, this error is not very big at each time step, because there is not very much extra mesh (no more than 10) in this example, while the total number of elements

DOI: 10.1002/fld

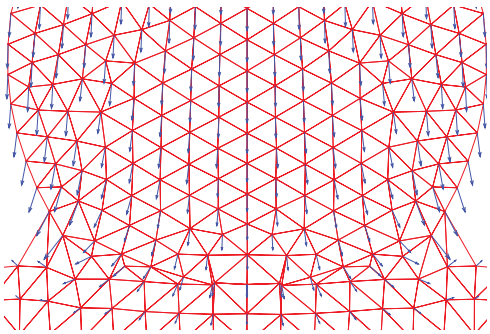

Figure 8. The mesh with fixed time step 0.01 s.

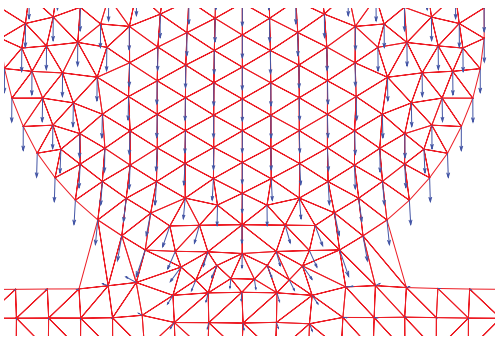

Figure 9. The mesh with adaptive time step.

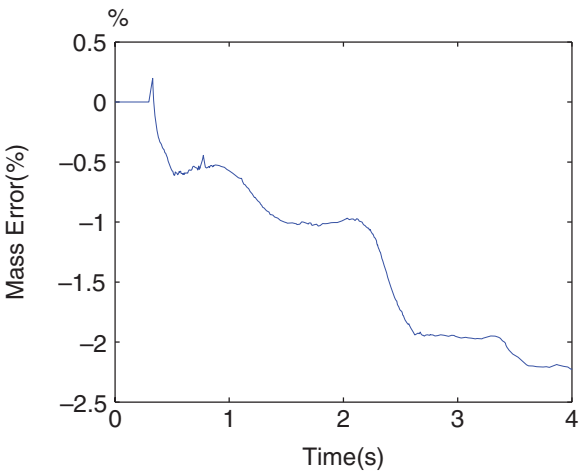

Figure 10. The mass error of the original PFEM.

DOI: 10.1002/fld

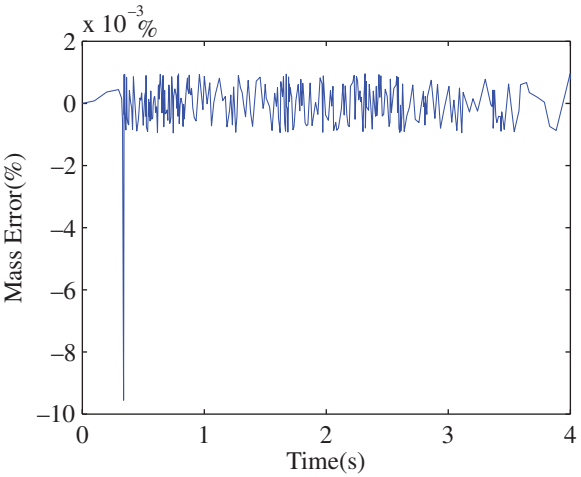

Figure 11. The mass error of the origin PFEM.

is about 5500. This means that the maximum error is about 0.2% at each time step. Futhermore, the error will not be accurate because this error is random; when the impact occurs the number of elements increases, and the number of elements decreases when the splash occurs.

The second aspect is that the mass will not be conserved for a *Lagrangian description* [22] and it will accumulate during the calculation.

Figure 10 illustrates the error of the mass during the calculation without any special treatment. We found that the mass increased about 0.2% when the circular drop impacted the water. Afterwards, the mass became small, which was very similar to the results in [21] and the error was about -2.5% at time 4 s. Figure 11 illustrates the error of the mass when the mesh-push method was used. First, the order of magnitude was about 10^{-5} most of the time but there were some special points with large negative errors. This is because the mesh-push algorithm conflicts with the α-shape procedure. But as mentioned before, this error is random and will not accumulate.

5.2.3. The velocity and pressure fields by three time-step methods. To study the influence of the time-step method, three time-step methods were applied and are illustrated in Figures 12 and 13. The velocity and pressure values at the center of the pool were obtained and the comparison is illustrated in Figure 12. We can see that the velocity obtained by BE becomes increasingly smaller; that is, the solution has been over dissipated. The velocity obtained by CN hardly oscillates. But the situation is not so fortunate for the pressure field. Figure 13 shows a comparison of the pressure fields. From Figure 13(b) we find that the pressure oscillates strongly because of the rough boundary by PFEM that is why the literature chooses the over dissipated BE method. The pressure obtained by BE (Figure 13(a)) is very good except at some points. In fact, at the moment when the impact occurs, the impact and the change of the free surface will influence the overall pressure fields because of the incompressibility restriction. The BE smooths these rough boundaries quickly while the CN cannot, and for the modified method (Figure 13(c)), some oscillations still happen, but they are smoothed quickly just as in the BE method.

DOI: 10.1002/fld

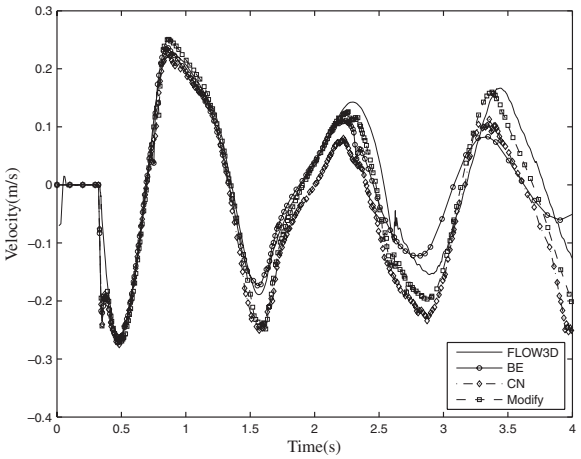

Figure 12. The *Y*-component of velocity by three time-step methods.

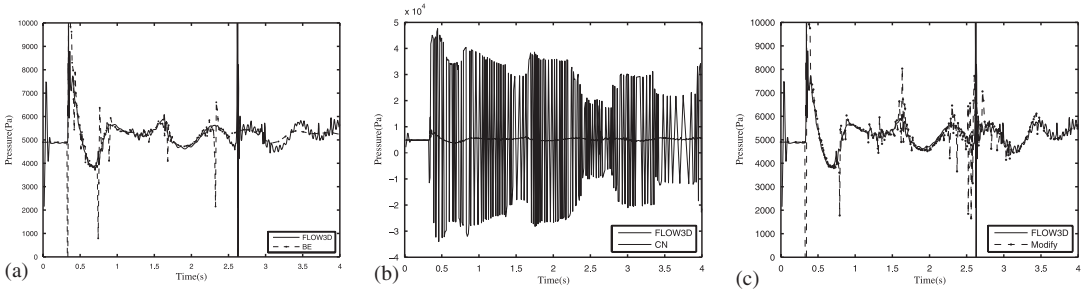

Figure 13. The pressure by three time-step methods.

5.2.4. Comparison with FLOW3D. With the modified pressure method, the shapes of the free surface are compared with FLOW3D at various times.

In the initial moment, the water drop begins to descend with the acceleration of gravity. Figure 14(a) shows the droplet impacting the pool's surface. At the moment shown in Figure 14(b), the fluids splash, and they impact the left and right walls as shown in Figure 14(c). The pool forms a big hole. In Figure 14(d), the hole has assembled slowly. When it reaches the highest point, as illustrated in Figure 14(e), the fluid at the center of the pool begins to push more fluid to its left and right sides and forms two peaks as in Figure 14(f). In Figure 14(g), part of the fluid sticks to the left and right walls. Because the velocity of the fluid is high at the center of the pool and is low at both the sides, a hat-shaped free surface is generated, as shown in Figure 14(h). The fluid moves down and up several times. As the range of the motion decreases, the collision of two traveling waves as shown in Figure 14(i) occurs. The motion becomes slower and slower and finally ceases due to viscosity.

Figure 15 shows the free surface shapes at some initial moments by FLOW3D. The results represented by solid symbols are from FLOW3D and the lines represent the free surface shape results from PFEM. The results are coincident with the modified PFEM. From it we can find

DOI: 10.1002/fld

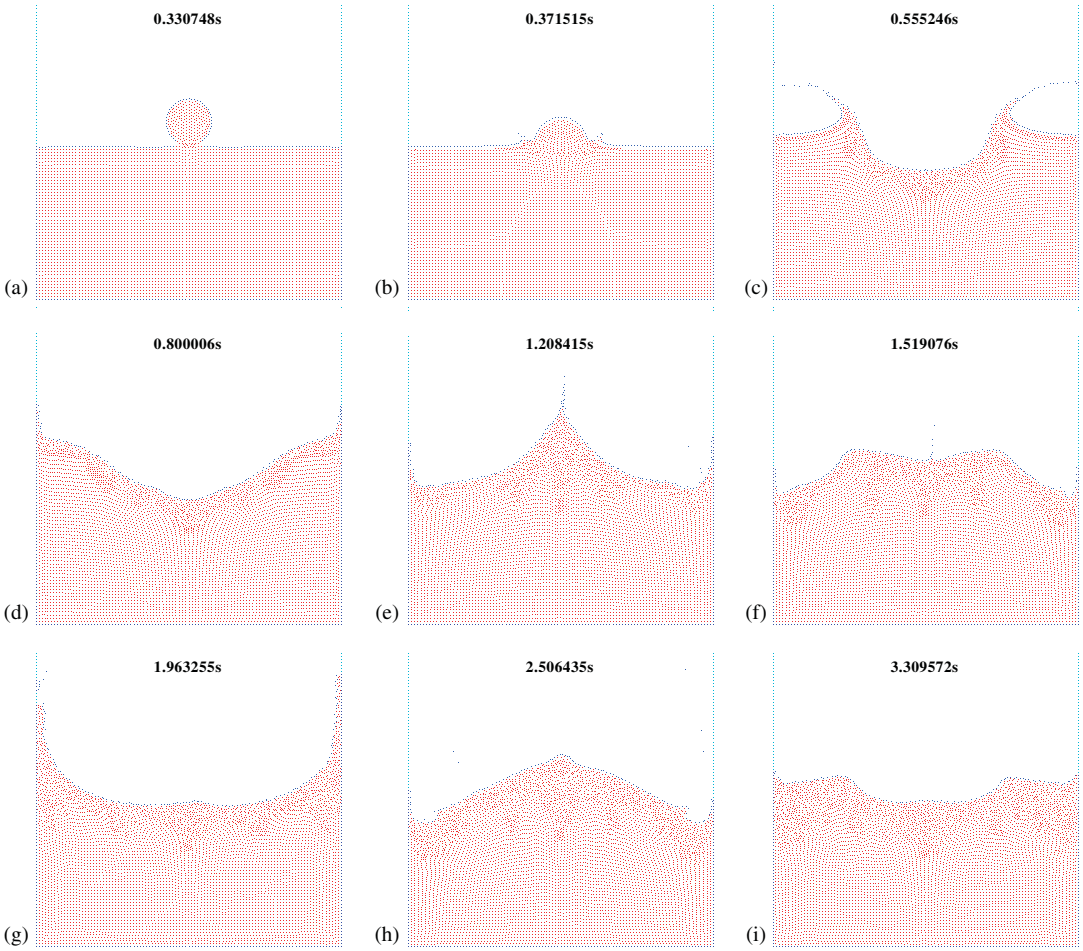

Figure 14. The shapes of free surface at various times.

that the water drops are circular according to both theory and PFEM, but the deformation takes place at the bottom of it according to FLOW3D. This is because of the imprecision of the surface reconstruction by the VOF method. However, under Lagrangian description, clear and sharp free surface shapes can always be obtained. Moreover, at time 2.5 s, the FLOW3D's results also have a hat-like free surface, although it is wider than the PFEM's. At time 3 s, the free surface by FLOW3D is not symmetric at all. Obviously, it is not very precise.

5.3. Sloshing in a tank

The forced sloshing of liquid in a tank subjected to horizontal base excitation was studied by PFEM. The tank was composed of two parts; the upper part was a rectangle with a width of 2 m and a height of 1 m, and the lower part was a semicircle with a radius of 1m (Figure 16). The motion of the fluid was accompanied with a time-varying body force $F = 0.2g\sin(4t)$, where $0.2g$ denotes the force amplitude, t is the time and 4 is the circular frequency of the force. Figure 17

DOI: 10.1002/fld

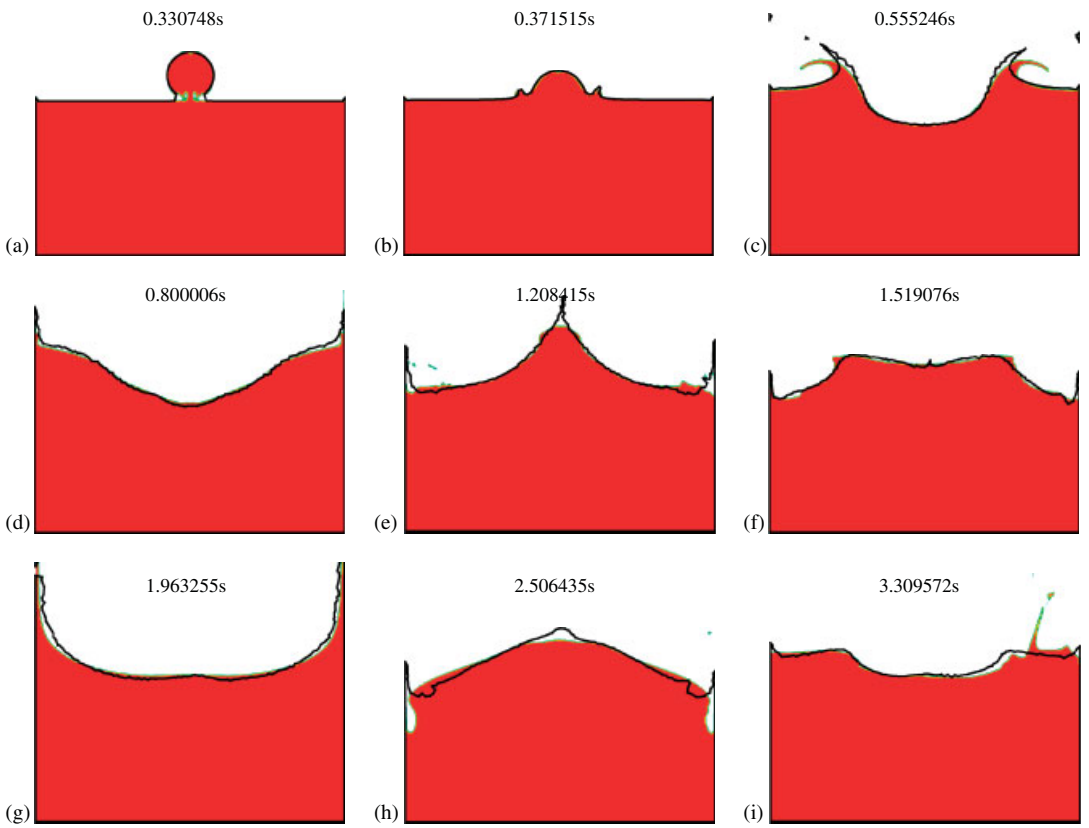

Figure 15. The comparison of the free surface by FLOW3D and PFEM.

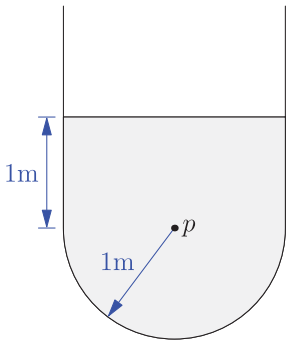

Figure 16. The sketch of the sloshing in a tank.

DOI: 10.1002/fld

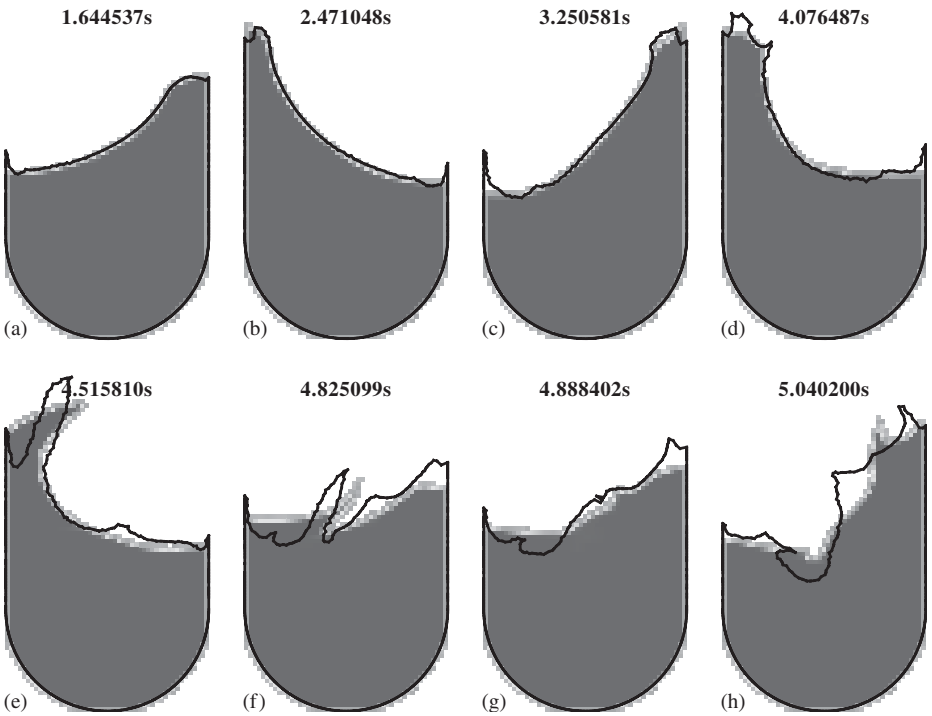

Figure 17. Large amplitude sloshing in the tank at different time steps.

shows the instantaneous configurations of the flow. The solid symbols represent the results from FLOW3D and the lines represent the free surface shape results from PFEM. In the first sloshing cycle, the fluid moves continuously as a whole. The amplitude of the sloshing is not very high, with its highest point near the wall of the tank (Figure 17(a)). After a cycle, this pattern is repeated, but the shape of the free surface is no longer as regular as in the first cycle (Figure 17(c)). In the first two cycles, the fluid splash and impact do not happen and this relatively simple case can be solved very well by the ALE method. But a more interesting case follows. With the excitation continuing, the amplitude of the sloshing increases. The pattern of the sloshing changes nonlinearly. The shape of the free surface is no longer regular, and begins to collapse (Figure 17(e)). Accompanied with such strong nonlinear phenomena, the velocities of the fluid at different positions become more diversified; some fluid particles are accelerated, and others are decelerated, which would cause breaking and splashing of the free surface, as illustrated in Figure 17(d) and (h), where the individual particles represent the splashed fluid. After a while, the collapsed fluid and the splashed fluid hit surrounding fluid and the impact occurs, as illustrated in Figure 17(f) and (g). PFEM can deal with impact without special treatment. The algorithm can capture the impact by generating a new linked mesh. The solution of the Navier–Stokes equation on this mesh will generate a greater pressure gradient to represent the impact.

To test the accuracy of PFEM, the pressure and velocity at the fixed position p as the center of the semicircle of the tank were compared with the commercial FLOW3D results as illustrated in Figures 18 and 19. In this example, the maximum mass error is -0.006% and the average time step is 0.0205 s.

DOI: 10.1002/fld

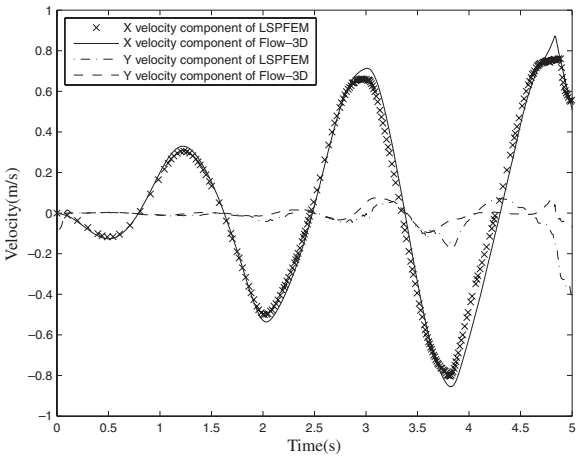

Figure 18. Computed velocity at the fixed position p.

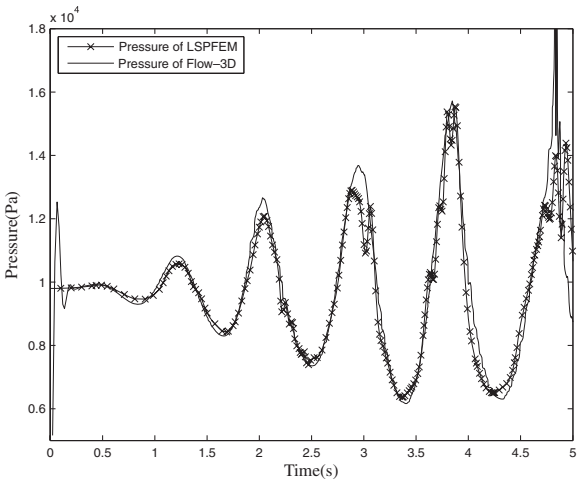

Figure 19. Computed pressure at the fixed position p.

5.4. Breaking of a dam with an obstacle

As shown in Figure 20, the initial water column with a width of 0.1461 m and a height of 0.292 m collapses in a region of 0.584 m × 0.584 m, and impacts a small obstacle at 0.0292 m. The obstacle has a width of 0.024 m and a height of 0.584 m. Figure 21 shows the mesh at 0.15 s for the adaptive time step and the method in [10]. Figure 22 shows a number of characteristic moments in the time evolution. In Figure 22(a), the water column impacts the obstacle. The fluid ejects vertically in Figure 22(b). In Figure 22(c), the fluid rushes over the obstacle and generates many breaking droplets. In Figure 22(d), the fluid impacts the right wall, and in Figure 22(e) and (f), most of the fluid arrives at the bottom of the region. At the time 0.38 s, the motion of the fluid by PFEM

DOI: 10.1002/fld

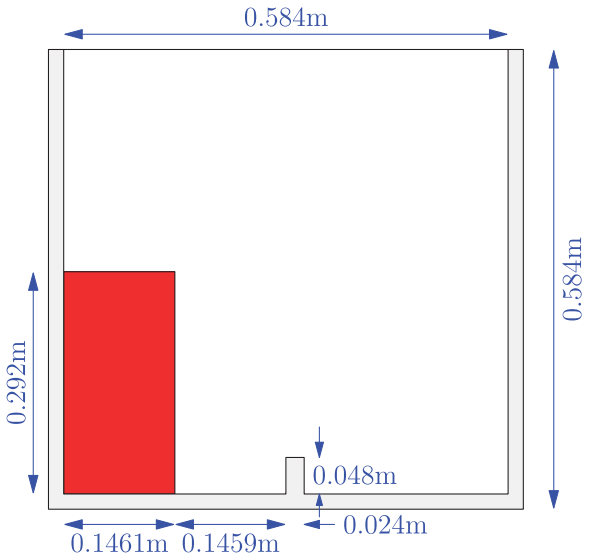

Figure 20. The sketch of the dam break problem.

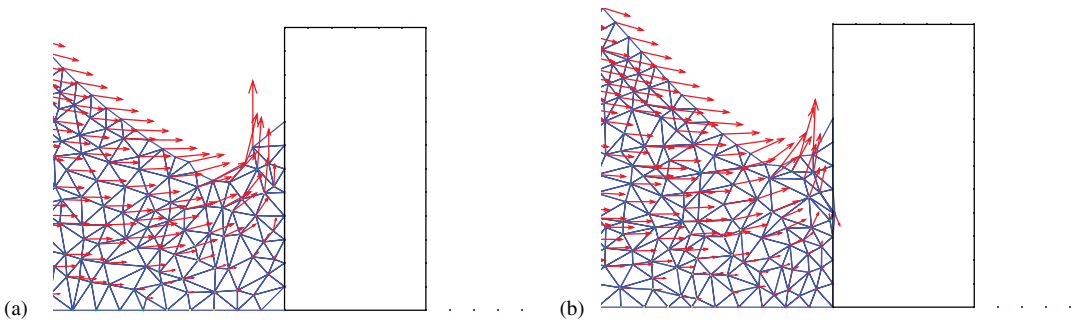

Figure 21. The mesh at 0.15 s by two kinds of time steps: (a) Time step posed here and (b) time step used in [10].

is about 0.02 s earlier than the result by FLOW3D. This phenomenon needs further investigation in the future.

In this example, the maximum mass error is -0.07% and the average time step is 0.0026 s.

Since in this problem there is no obvious reference point, the velocity and pressure are not compared with FLOW3D here.

6. CONCLUSION

In this work, the PFEM was improved, and the relevant phenomena of droplet, sloshing and dam break were simulated. In these examples the shape of the free surface and the velocity and pressure fields were examined by the commercial FLOW3D code.

DOI: 10.1002/fld

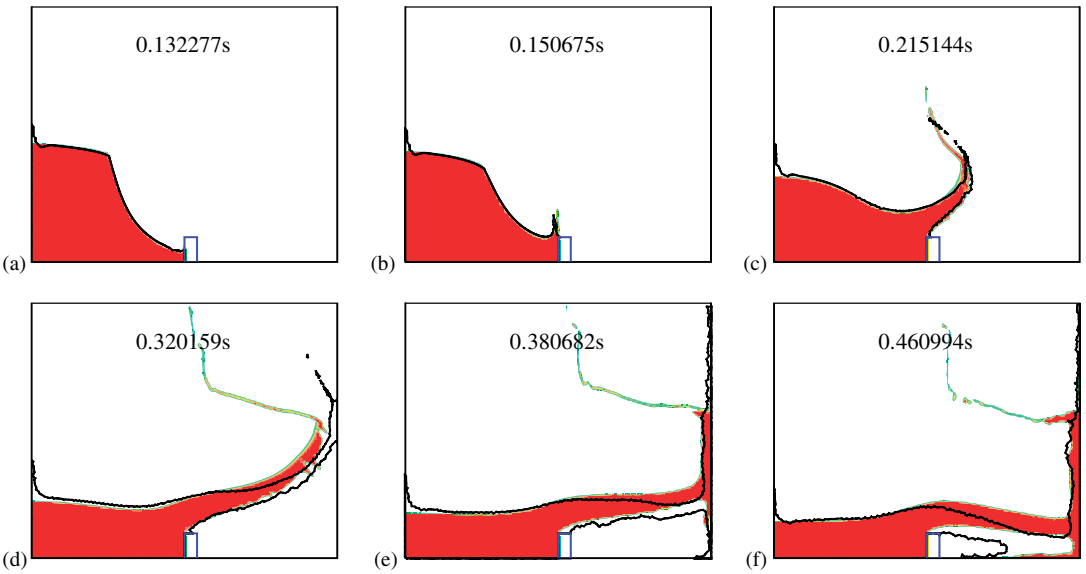

Figure 22. Dam break against an obstacle.

The improvements covered four aspects. (1) The least-square finite element method (LSFEM) was used as a substitute for the original finite calculus (FIC) method. This substitution is not necessary, but it can simplify the programming difficulties greatly, because the stabilization terms needed by FIC are unnecessary in the LSFEM. As we all know, the stabilization terms are tedious because they sometimes depend on experiences. (2) A new adaptive time step was derived to improve the efficiency of and to overcome the abundant 'layers of nodes' of the original PFEM. (3) A mesh-pull procedure was adopted to reduce the relative mass error to 10^{-5}, while the mass conservation was not very good for PFEM. (4) A method to decrease the pressure oscillations was created to balance the oscillation by the CN method and the large numerical dissipation generated by the backward Euler (BE) method, and this dissipated BE method was used in the original PFEM.

ACKNOWLEDGEMENTS

This research was supported by the National Natural Science Foundation (10302013, 10572022). The authors are grateful to Dr Bo-nan Jiang for his fruitful discussions on some issues about LSFEM.

REFERENCES

1. Scardovelli R, Zaleski S. Direct numerical simulation of free-surface and interfacial flow. *Annual Review of Fluid Mechanics* 1999; **31**(1):567–603.
2. Hirt CW, Nichols BD. Volume of fluid (VOF) method for the dynamics of free boundaries. *Journal of Computational Physics* 1981; **39**(1):201–225.
3. Sussman M, Smereka P, Osher S. A level set approach for computing solutions to incompressible two-phase flow. *Journal of Computational Physics* 1994; **114**(1):146–159.
4. Monaghan JJ. Smoothed particle hydrodynamics. *Reports on Progress in Physics* 2005; **68**(8):1703–1759.

DOI: 10.1002/fld

5. Belytschko T, Lu YY, Gu L. Element-free Galerkin methods. *International Journal for Numerical Methods in Engineering* 1994; **37**(2):229–256.
6. González D, Cueto E, Chinesta F, Doblaré M. A natural element updated Lagrangian strategy for free-surface fluid dynamics. *Journal of Computational Physics* 2007; **223**(1):127–150.
7. Onate E, Idelsohn SR, Del Pin F, Aubry R. The particle finite element method—an overview. *International Journal of Computational Methods* 2004; **1**(2):267–307.
8. Oñate E. Possibilities of finite calculus in computational mechanics. *International Journal for Numerical Methods in Engineering* 2004; **60**(1):255–281.
9. Oñate E, Valls A, García J. FIC/FEM formulation with matrix stabilizing terms for incompressible flows at low and high Reynolds numbers. *Computational Mechanics* 2006; **38**(4):440–455.
10. Oñate E, García J, Idelsohn SR, Pin FD. Finite calculus formulations for finite element analysis of incompressible flows. Eulerian, ALE and Lagrangian approaches. *Computer Methods in Applied Mechanics and Engineering* 2006; **195**(23–24):3001–3037.
11. Jiang BN. *The Least-Squares Finite Element Method*: *Theory and Applications in Computational Fluid Dynamics and Electromagnetics*. Springer: New York, 1998.
12. Idelsohn SR, Calvo N, Oñate E. Polyhedrization of an arbitrary 3D point set. *Computer Methods in Applied Mechanics and Engineering* 2003; **192**(22–24):2649–2667.
13. Bowyer A. Computing Dirichlet tessellations. *The Computer Journal* 1981; **24**(2):162–166.
14. Sukumar N, Moran B, Semenov AY, Belikov VV. Natural neighbour Galerkin methods. *International Journal for Numerical Methods in Engineering* 2001; **50**(1):1–27.
15. Idelsohn SR, Onate E, Calvo N, Del Pin F. The meshless finite element method. *International Journal for Numerical Methods in Engineering* 2003; **58**:893–912.
16. Gonzalez D, Cueto E, Martinez M. Numerical integration in natural neighbour Galerkin methods. *International Journal for Numerical Methods in Engineering* 2004; **60**(12):2077–2104.
17. Edelsbrunner H, Mücke EP. *Three-dimensional Alpha Shapes*. ACM Press: New York, NY, U.S.A., 1992.
18. Kayser-Herold O. Least-squares methods for the solution of fluid–structure interaction problems. *Ph.D. Thesis*, Institut für Wissenschaftliches Rechnen, TU Braunschweig, 2006.
19. Chang CL, Nelson JJ. Least-squares finite element method for the Stokes problem with zero residual of mass conservation. *SIAM Journal on Numerical Analysis* 1997; **34**(2):480–489.
20. Deang JM, Gunzburger MD. Issues related to least-squares finite element methods for the Stokes equations. *SIAM Journal on Scientific Computing* 1998; **20**(3):878–906.
21. Birknes J, Pedersen G. A particle finite element method applied to long wave run-up. *International Journal for Numerical Methods in Fluids* 2006; **52**(3):237–261.
22. Yamamoto K, Kawahara M. Structural oscillation control using tuned liquid damper. *Computers and Structures* 1999; **71**(4):435–446.
23. Idelsohn SR, Oñate E. To mesh or not to mesh. That is the question. *Computer Methods in Applied Mechanics and Engineering* 2006; **195**(37–40):4681–4696.

DOI: 10.1002/fld