INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng 2004; **61**:964–989 (DOI: 10.1002/nme.1096)

The particle finite element method: a powerful tool to solve incompressible flows with free-surfaces and breaking waves

S. R. Idelsohn[1,2,*,†], E. Oñate[2] and F. Del Pin[1]

[1]*International Center for Computational Methods in Engineering (CIMEC), Universidad Nacional del Litoral and CONICET, Santa Fe, Argentina*
[2]*International Center for Numerical Methods in Engineering (CIMNE), Universidad Politécnica de Cataluña, Barcelona, Spain*

SUMMARY

Particle Methods are those in which the problem is represented by a discrete number of particles. Each particle moves accordingly with its own mass and the external/internal forces applied to it. Particle Methods may be used for both, discrete and continuous problems. In this paper, a Particle Method is used to solve the continuous fluid mechanics equations. To evaluate the external applied forces on each particle, the incompressible Navier–Stokes equations using a Lagrangian formulation are solved at each time step. The interpolation functions are those used in the Meshless Finite Element Method and the time integration is introduced by an implicit fractional-step method. In this manner classical stabilization terms used in the momentum equations are unnecessary due to lack of convective terms in the Lagrangian formulation. Once the forces are evaluated, the particles move independently of the mesh. All the information is transmitted by the particles. Fluid–structure interaction problems including free-fluid-surfaces, breaking waves and fluid particle separation may be easily solved with this methodology. Copyright © 2004 John Wiley & Sons, Ltd.

KEY WORDS: particle methods; finite element methods; fractional step; lagrange formulations; incompressible Navier–Stokes equations; implicit time integration; fluid–structure interactions; free-surfaces; breaking waves

1. INTRODUCTION

Over the last 20 years, computer simulation of incompressible fluid flow has been based on the Eulerian formulation of the fluid mechanics equations on continuous domains. However, it is still difficult to analyse problems in which the shape of the interface changes continuously

*Correspondence to: S. R. Idelsohn, International Center for Computational Methods in Engineering (CIMEC), Universidad Nacional del Litoral and CONICET, Santa Fe, Argentina.
†E-mail: sergio@ceride.gov.ar

Received 11 July 2003
Revised 8 January 2004
Accepted 13 February 2004

or in fluid–structure interactions with free-surfaces where complicated contact problems are involved.

More recently, *Particle Methods* in which each fluid particle is followed in a Lagrangian manner have been used [1–4]. The first ideas on this approach were proposed by Monaghan [1] for the treatment of astrophysical hydrodynamic problems with the so-called *Smooth Particle Hydrodynamics Method* (SPH). This method was later generalized to fluid mechanic problems [2–4]. Kernel approximations are used in the SPH method to interpolate the unknowns. Within the family of Lagrangian formulations the Free Lagrange Method (FLM) [5, 6] received a lot of attention throughout 1980s. Basically the FLM is an adaptation of the finite volume method in a Lagrangian scheme that uses the Voronoi diagram of freely moving points to partition the domain. As a drawback, we might say that poor aspect-ratio Voronoi cells results in poor resolution of the final results.

On the other hand, a family of methods called *Meshless Methods* have been developed both for structural [7–9] and fluid mechanics problems [10–13]. All these methods use the idea of a polynomial interpolant that fits a number of points minimizing the distance between the interpolated function and the value of the unknown point. These ideas were proposed first by Nayroles *et al.* [9] which were later used in structural mechanics by Belytschko *et al.* [7] and in fluid mechanics problems by Oñate *et al.* [10–13]. In a previous paper, the authors presented the numerical solution for the fluid mechanics equations using a Lagrangian formulation and a meshless method called the *Finite Point Method* (FPM) [10]. Lately, the meshless ideas were generalized to take into account the finite element type approximations in order to obtain the same computing time in mesh generation as in the evaluation of the meshless connectivities [13]. This method was called the *Meshless Finite Element Method* (MFEM) and uses the extended Delaunay tessellation [14] to build a mesh combining elements of different polygonal (or polyhedral in 3D) shapes in a computing time which is linear with the number of nodal points.

It must be noted that particle methods may be used with either mesh-based or meshless shape functions. The only practical limitation is that the connectivities in meshless methods or the mesh generation in mesh-based methods need to be evaluated at each time step.

In this paper, a particle method will be used together with a particular form of the FEM. The new method will be called the Particle Finite Element Method (PFEM). To evaluate the forces on each particle the incompressible Navier–Stokes equations on a continuous domain will be solved using the MFEM shape functions [13] in space. Those functions are generated in a computing time order 'n' where 'n' being the number of particles. From the computing time point of view, this is the same (or even better) than the computing time to evaluate the connectivities in a meshless method. Furthermore, the shape functions proposed by the MFEM have big advantages compared with those obtained via any other meshless method: all the classical advantages of the FEM for the evaluation of the integrals of the unknown functions and their derivatives are preserved, including the facilities to impose the boundary conditions and the use of symmetric Galerkin approximations.

The Lagrangian fluid flow equations for the Navier–Stokes approximation will be revised in the next sections including an implicit fractional-step method for the time integration. Then, the particle method proposed will be used to solve some FSI problems with rigid solids and fluid flows including free-surfaces and breaking waves.

2. PARTICLE METHODS

Particle Methods aim to represent the behaviour of a physical problem by a collection of particles. Each particle moves accordingly with its own mass and the internal/external forces applied on it. External forces are evaluated by the interaction with the neighbour particles by simple rules.

A particle may be a physical part of the domain (spheres, rocks, powder, etc.) or a specific part of the continuous domain previously defined.

Another characteristic of Particle Methods is that all the physical and mathematical properties are attached to the particle itself and not to the elements as in the FEM. For instance, physical properties like viscosity or density, physical variables like velocity, temperature or pressure and also mathematical variables like gradients or volumetric deformations are assigned to each particle and they represent an average of the property around the particle position.

Particle methods are advantageous to treat discrete problems like granular materials but also to treat continuous problems in which there are possibilities of internal separations, contact problems or free-surfaces with breaking waves.

Accordingly to the way to evaluate the forces applied to each particle, the method may be divided into two categories: those in which the interacting forces between the particles are evaluated by a local contact problem [15] and those in which the forces are evaluated by solving a continuous differential equation in the entire domain [16]. This paper concerns with the last category.

Finally, the most crucial characteristic of a Particle Method is that there is not a specified solution domain. The problem domain is defined by the particle positions and hence, there is not a boundary surface or line. This is the reason why, when a differential equation is to be solved in order to evaluate the forces, the boundary surface needs to be identified in order to impose the boundary conditions. In addition, the particles can be used to generate a discrete domain within which the integral form of the governing differential equations is solved (see Figure 1).

In this paper, a Particle Finite Element Method is proposed to deal with the incompressible Navier–Stokes equations. Then, the true material will be continuous and incompressible when it is submitted to compression forces, but with the possibility to separate under traction forces. This is the case of most physical fluids, like water, oils and other fluids with low rate of surface tractions.

Both, fluid and solid materials will be modelled by an arbitrary number of particles. On each particle the acting forces will be the gravity force (internal force of the particle) and the interacting forces with the neighbour particles (external force to the particle). The external forces will be evaluated solving the Navier–Stokes equations. For this reason a domain needs to be defined at each time step with a defined boundary surface where the boundary conditions will be imposed. Also at each time step a new mesh is generated in order to define shape functions to solve the differential equations. This mesh is only useful for the definition of the interacting forces and vanishes once the forces are evaluated (see Figure 1). The interpolation functions to be used are a particular case of the Finite Element Method shape functions. The boundary surface is defined using the Alpha-Shape Method explained in Section 4. The evaluation of the interacting forces between particles is described next.

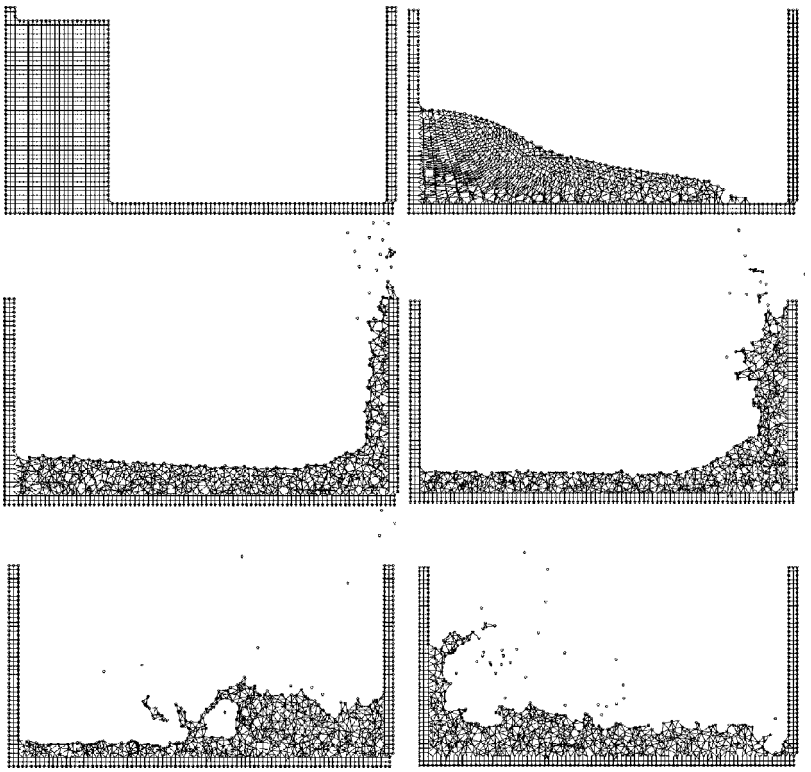

Figure 1. Recognition of the boundary of the analysis domain and mesh update for successive point distributions.

3. PARTICLE POSITION UPDATE

The particle positions will be updated via solving the Lagrangian form of the Navier–Stokes equations.

Let X_i be the initial position of a particle at time t^n. Let x_i the final position of a particle at time t^{n+1} and the time increment $\Delta t = t^{n+1} - t^n$ and $u_i(x, t^{n+1}) = u_i^{n+1}$ being the velocity of the particle at time t^{n+1}, the final position can be approximated by

$$x_i = X_i + u_i^{n+1} \Delta t$$

In the same way the displacement of the particle $d_i(x, t^{n+1}) = d_i^{n+1} = u_i^{n+1} \Delta t$.

3.1. Governing Lagrangian equations in a viscous fluid flow

In the final x_i position, the mass and momentum conservation equations can be written as

Mass conservation:

$$\frac{\mathrm{D}\rho}{\mathrm{D}t} + \rho \frac{\partial u_i}{\partial x_i} = 0 \tag{1}$$

Momentum conservation:

$$\rho \frac{\mathrm{D}u_i}{\mathrm{D}t} = -\frac{\partial}{\partial x_i} p + \frac{\partial}{\partial x_j} \tau_{ij} + \rho f_i \tag{2}$$

where ρ is the density u_i are the Cartesian components of the velocity field, p the pressure, τ_{ij} the deviator stress tensor, f_i the source term (normally the gravity) and $\mathrm{D}\phi/\mathrm{D}t$ represents the total or material time derivative of a function ϕ.

For Newtonian fluids the stress tensor τ_{ij} may be expressed as a function of the velocity field through the viscosity μ by

$$\tau_{ij} = \mu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{2}{3} \frac{\partial u_l}{\partial x_l} \delta_{ij} \right) \tag{3}$$

For near incompressible flows ($\partial u_i/\partial x_i \ll \partial u_k/\partial x_l$) the term

$$\frac{2\mu}{3} \frac{\partial u_i}{\partial x_i} \approx 0 \tag{4}$$

and it may be neglected in Equation (3). Then

$$\tau_{ij} \approx \mu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \tag{5}$$

In the same way, the term $(\partial/\partial x_j)\tau_{ij}$ in the momentum equations may be simplified for near incompressible flows as

$$\begin{aligned} \frac{\partial}{\partial x_j} \tau_{ij} &= \frac{\partial}{\partial x_j} \left(\mu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \right) = \mu \frac{\partial}{\partial x_j} \left(\frac{\partial u_i}{\partial x_j} \right) + \mu \frac{\partial}{\partial x_j} \left(\frac{\partial u_j}{\partial x_i} \right) \\ &= \mu \frac{\partial}{\partial x_j} \left(\frac{\partial u_i}{\partial x_j} \right) + \mu \frac{\partial}{\partial x_i} \left(\frac{\partial u_j}{\partial x_j} \right) \approx \mu \frac{\partial}{\partial x_j} \left(\frac{\partial u_i}{\partial x_j} \right) \end{aligned} \tag{6}$$

Then, the momentum equations can be finally written as

$$\rho \frac{\mathrm{D}u_i}{\mathrm{D}t} = -\frac{\partial}{\partial x_i} p + \frac{\partial}{\partial x_j} \tau_{ij} + \rho f_i \approx -\frac{\partial}{\partial x_i} p + \mu \frac{\partial}{\partial x_j} \left(\frac{\partial u_i}{\partial x_j} \right) + \rho f_i \tag{7}$$

Boundary conditions: On the boundaries, the standard boundary conditions for the Navier–Stokes equations are

$$\begin{aligned} \tau_{ij} v_j - p v_i &= \bar{\sigma}_{ni} \quad \text{on } \Gamma_\sigma \\ u_i v_i &= \bar{u}_n \quad \text{on } \Gamma_n \\ u_i \zeta_i &= \bar{u}_t \quad \text{on } \Gamma_t \end{aligned}$$

where v_i and ζ_i are the components of the normal and tangent vector to the boundary.

3.2. Implicit–explicit time integration

Equation (7) will be integrated implicitly in time as

$$\rho \frac{\mathrm{D}u_i}{\mathrm{D}t} \approx \rho \frac{\mathrm{u}_i(x_i, t^{n+1}) - u_i(X_i, t^n)}{\Delta t} = \rho \frac{u_i^{n+1} - u_i^n}{\Delta t} = \left[-\frac{\partial}{\partial x_i} p + \mu \frac{\partial}{\partial x_j} \left(\frac{\partial u_i}{\partial x_j} \right) + \rho f_i \right]^{n+\theta} \tag{8}$$

where $[\phi(x,t)]^{n+\theta}$ means $\theta\phi(x,t^{n+1}) + (1-\theta)\phi(x,t^n) = \theta\phi^{n+1} + (1-\theta)\hat{\phi}^n$ and $\hat{\phi}^n = \phi(x,t^n)$ represents the value of the function at time t^n but at the final position x. For simplicity ϕ^n will be used instead of $\hat{\phi}^n$.

Only the case of $\theta = 1$ (full implicit) will be considered next. Other values, as for instance $\theta = \frac{1}{2}$, may be considered without major changes.

The time integrated equations become

$$\rho \frac{u_i^{n+1} - u_i^n}{\Delta t} = \left[-\frac{\partial}{\partial x_i} p \right]^{n+1} + \left[\mu \frac{\partial}{\partial x_j} \left(\frac{\partial u_i}{\partial x_j} \right) + \rho f_i \right]^{n+1} \tag{9}$$

The mass conservation is also integrated implicitly by

$$\frac{\mathrm{D}\rho}{\mathrm{D}t} \approx \frac{\rho^{n+1} - \rho^n}{\Delta t} = -\rho^{n+1} \frac{\partial (u_i^{n+1})}{\partial x_i} \tag{10}$$

3.3. The time splitting

The time integration of Equations (9) presents some difficulties because it is a fully coupled equation involving four degrees of freedom by node. When the fluid is incompressible or nearly incompressible, advantages can be taken from the fact that in Equations (9) the three components of the velocity are only coupled via the pressure. The fractional-step method proposed in Reference [17] will be used. This basically consists in splitting each time step into two pseudo-time steps. In the first step, the implicit part of the pressure is avoided in order to have a decoupled equation in each of the velocity components. The implicit part of the pressure is added during a second step. The fractional-step algorithm for Equations (9) and (10) is the following:

Split of the momentum equations

$$\frac{\mathrm{D}u_i}{\mathrm{D}t} \approx \frac{u_i^{n+1} - u_i^n}{\Delta t} = \frac{u_i^{n+1} - u_i^* + u_i^* - u_i^n}{\Delta t} = -\frac{1}{\rho} \frac{\partial}{\partial x_i} p^{n+1} + \frac{1}{\rho} \frac{\partial \tau_{ij}^{n+\theta}}{\partial x_j} + f_i \tag{11}$$

where u_i^* are fictitious variables termed fractional velocities defined by the split

$$\text{(A)} \quad u_i^* = u_i^n + f_i \Delta t - \frac{\Delta t}{\rho} \frac{\partial}{\partial x_i} \gamma p^n + \frac{\Delta t}{\rho} \frac{\partial}{\partial x_j} \tau_{ij}^{n+\theta} \tag{12}$$

$$\text{(C)} \quad u_i^{n+1} = u_i^* - \frac{\Delta t}{\rho} \frac{\partial}{\partial x_i} (p^{n+1} - \gamma p^n) \tag{13}$$

in which $p^n = p(x, t^n)$ is the value of the pressure at time t^n but evaluated at the final position and f_i is considered constant in time.

In Equations (12) and (13) γ is a parameter giving the amount of pressure splitting, varying between 0 and 1. A larger value of γ means small pressure split. In this paper γ will be fixed to 0 in order to have the larger pressure split and hence, a better pressure stabilization. Other values as, for instance $\gamma = 1$, may be used to derive high-order schemes in time [17].

Taking into account (6), the last term in (12) may be written as

$$\frac{\partial}{\partial x_j}\tau_{ij}^{n+\theta} = \mu\frac{\partial}{\partial x_j}\left(\frac{\partial u_i^{n+\theta}}{\partial x_j}\right) = \mu(1-\theta)\frac{\partial}{\partial x_j}\left(\frac{\partial \hat{u}_i^n}{\partial x_j}\right) + \mu\theta\frac{\partial}{\partial x_j}\left(\frac{\partial u_i^{n+1}}{\partial x_j}\right)$$

The following approximations have been introduced [17]:

$$\mu\frac{\partial}{\partial x_j}\left(\frac{\partial u_i^{n+\theta}}{\partial x_j}\right) \approx \mu(1-\theta)\frac{\partial}{\partial x_j}\left(\frac{\partial \hat{u}_i^n}{\partial x_j}\right) + \mu\theta\frac{\partial}{\partial x_j}\left(\frac{\partial u_i^*}{\partial x_j}\right)$$

This allows to write Equation (12) as

$$u_i^* = u_i^n + f_i\ \Delta t - \frac{\Delta t}{\rho}\frac{\partial}{\partial x_i}\gamma\hat{p}^n + \frac{\Delta t}{\rho}\mu(1-\theta)\frac{\partial}{\partial x_j}\left(\frac{\partial \hat{u}_i^n}{\partial x_j}\right) + \frac{\Delta t}{\rho}\mu\theta\frac{\partial}{\partial x_j}\left(\frac{\partial u_i^*}{\partial x_j}\right)$$

For $\gamma = 1$ and $\theta = 1$

$$u_i^* - \frac{\Delta t}{\rho}\mu\frac{\partial}{\partial x_j}\left(\frac{\partial u_i^*}{\partial x_j}\right) = u_i^n + f_i\ \Delta t \tag{14}$$

Split of the mass conservation equations

$$\frac{\mathrm{D}\rho}{\mathrm{D}t} \approx \frac{\rho^{n+1}-\rho^n}{\Delta t} = \frac{\rho^{n+1}-\rho^*+\rho^*-\rho^n}{\Delta t} = -\rho\frac{\partial(u_i^{n+1}-u_i^*+u_i^*)}{\partial x_i} \tag{15}$$

where ρ^* is a fictitious variable defined by the split

$$\frac{\rho^*-\rho^n}{\Delta t} = -\rho\frac{\partial u_i^*}{\partial x_i} \tag{16a}$$

$$\frac{\rho^{n+1}-\rho^*}{\Delta t} = -\rho\frac{\partial(u_i^{n+1}-u_i^*)}{\partial x_i} \tag{16b}$$

Coupled equations

From Equations (13) and (16) the coupled mass–momentum equation becomes

$$\text{(B)}\quad \frac{\rho^{n+1}-\rho^*}{\Delta t^2} = \frac{\partial^2}{\partial x^2}(p^{n+1}) \tag{17}$$

Taking into account Equation (16a), the above expression can be written as

$$\frac{\rho^{n+1}-\rho^n}{\Delta t^2} + \frac{\rho}{\Delta t}\frac{\partial u_i^*}{\partial x_i} = \frac{\partial^2}{\partial x_i^2}(p^{n+1}) \tag{18}$$

In Equation (18) the incompressibility condition has not been introduced yet. The simplest way to introduce the incompressibility condition in a Lagrangian formulation is to write

$$\rho^{n+1} = \rho^n = \rho^0 = \rho \tag{19}$$

Then, the first term of Equation (18) disappears, giving

$$\frac{\rho}{\Delta t}\frac{\partial u_i^*}{\partial x_i} = \frac{\partial^2}{\partial x_i^2}(p^{n+1})$$

The three step fractional method used here can be summarized by

$$\begin{aligned}
&\text{(A)}\quad u_i^* - \frac{\Delta t}{\rho}\mu\frac{\partial}{\partial x_j}\left(\frac{\partial u_i^*}{\partial x_j}\right) = u_i^n + f_i\Delta t \quad \Rightarrow \quad u_i^* \\
&\text{(B)}\quad \frac{\rho}{\Delta t}\frac{\partial u_i^*}{\partial x_i} = \frac{\partial^2}{\partial x_i^2}(p^{n+1}) \quad \Rightarrow \quad p^{n+1} \\
&\text{(C)}\quad u_i^{n+1} = u_i^* - \frac{\Delta t}{\rho}\frac{\partial}{\partial x_i}(p^{n+1}) \quad \Rightarrow \quad u_i^{n+1}
\end{aligned} \tag{20}$$

3.4. Generation of a new mesh

One of the key points for the success of the Lagrangian flow formulation described here is the fast regeneration of a mesh at every time step on the basis of the position of the nodes in the space domain. In this work, the mesh is generated using the so-called extended Delaunay tesselation (EDT) presented in Reference [14]. The EDT allows to generate meshes of elements with arbitrary polyhedrical shapes (combining triangles, quadrilaterals and other polygons in 2D and tetrahedra, hexahedra and arbitrary polyhedra in 3D) in a computing time of order n, n being the total number of nodes in the mesh.

The shape functions for arbitrary polyhedral elements can be simply obtained using the so-called non-sibsonian interpolations [18]. Details of the mesh generation procedure and the shape functions for arbitrary polyhedra can be found in References [13, 14].

Once the new mesh has been generated at each time step the numerical solution is found using the finite element algorithm described in the paper. The combination of elements with different geometrical shapes in the same mesh is one of the innovative aspects of the Lagrangian formulation presented here.

3.5. Spatial discretization via the Meshless Finite Element Method (MFEM)

The unknown functions are approximated using an equal order interpolation for all variables in the final configuration

$$u_i = \sum_l N_l(X,t)U_{il}$$

$$p = \sum_l N_l(X,t)P_l$$

In matrix form

$$u_i = \mathbf{N}^{\mathrm{T}}(X,t)\mathbf{U}_i$$
$$p = \mathbf{N}^{\mathrm{T}}(X,t)\mathbf{P} \tag{21}$$

or in compact form

$$u_i = \mathbf{N}_i^{\mathrm{T}}\mathbf{U} = \begin{bmatrix} \mathbf{N}^{\mathrm{T}} & & \\ & \mathbf{N}^{\mathrm{T}} & \\ & & \mathbf{N}^{\mathrm{T}} \end{bmatrix} \mathbf{U} \tag{22}$$

where $\mathbf{N}^{\mathrm{T}}$ are the MFEM shape functions and $\mathbf{U}, \mathbf{P}$ the nodal values of the three components of the unknown velocity and the pressure, respectively.

It must be noted that the shape functions $\mathbf{N}(X,t)$ are functions of the particle co-ordinates. Then, the shape functions may change in time following the particles position. During the time step a mesh update may introduce change in the shape function definition which must be taken into account. During the time integration there are two times involved: t^n and t^{n+1}. The following notation will be used to distinguish between $\mathbf{N}(X,t^n)$ and $\mathbf{N}(X,t^{n+1})$:

$$\mathbf{N}(X,t^n) = \mathbf{N}^n \quad \text{and} \quad \mathbf{N}(X,t^{n+1}) = \mathbf{N}^{n+1} \tag{23}$$

Nevertheless, the following hypothesis will be introduced: There is no mesh update during each time step. This means that if a mesh update is introduced at the beginning of a time step, the same mesh (but deformed) will be kept until the end of the time step.

Mathematically this means

$$\mathbf{N}(X,t^n) = \mathbf{N}(X,t^{n+1}) \tag{24}$$

Unfortunately, this hypothesis is not always possible to satisfy for all meshes and thus introduces small errors in the computation which are neglected in this paper.

Using the Galerkin weighted residual method to solve the split equations the following integrals must be written:

$$\text{(A)} \quad \int_V \mathbf{N}_i u_i^* \,\mathrm{d}V \,\frac{\rho}{\Delta t} - \int_V \mathbf{N}_i u_i^n \,\mathrm{d}V \,\frac{\rho}{\Delta t} - \int_V \mathbf{N}_i f_i \rho \,\mathrm{d}V + \int_V \mathbf{N}_i \,\frac{\partial}{\partial x_i}\gamma p^n \,\mathrm{d}V + \int_V \mathbf{N}_i \mu \,\frac{\partial}{\partial x_j}\left\{\left(\frac{\partial u_i^{n+\theta}}{\partial x_j}\right)\right\}\mathrm{d}V - \int_{\Gamma_\sigma} \mathbf{N}_i(\bar{\sigma}_{ni} - (\tau_{ij}^{n+\theta} v_j - \gamma p^n v_i))\,\mathrm{d}\Gamma = 0 \tag{25}$$

$$\text{(B)} \quad \int_V \mathbf{N}\left\{\frac{\rho}{\Delta t}\left(\frac{\partial u_i^*}{\partial x_i}\right) - \frac{\partial^2}{\partial x_i^2}(p^{n+1} - \gamma p^n)\right\}\mathrm{d}V + \frac{\rho}{\Delta t}\int_{\Gamma u} \mathbf{N}(\bar{u}_i^{n+1} v_i - u_i^{n+1} v_i)\,\mathrm{d}\Gamma = 0 \tag{26}$$

$$\text{(C)}\quad \int_V \mathbf{N}_i \left\{ (u_i^{n+1} - u_i^*)\,\frac{\rho}{\Delta t} + \frac{\partial}{\partial x_i}\,(p^{n+1} - \gamma p^n) \right\} \mathrm{d}V - \int_{\Gamma_\sigma} \mathbf{N}_i (p^{n+1} - \gamma \hat{p}^n) v_i \,\mathrm{d}\Gamma = 0 \tag{27}$$

where the boundary conditions have also been split and V is the volume at time t^{n+1}.

Integrating by parts some of the terms, the above equations become

$$\text{(A)}\quad \int_V \mathbf{N}_i (u_i^* - f_i \Delta t)\,\frac{\rho}{\Delta t}\,\mathrm{d}V - \int_V \mathbf{N}_i u_i^n\,\frac{\rho}{\Delta t}\,\mathrm{d}V + \int_V \mathbf{N}_i \frac{\partial}{\partial x_i}\,\gamma p^n\,\mathrm{d}V + \mu \int_V \frac{\partial \mathbf{N}_i}{\partial x_j}\,\frac{\partial u_i^{n+\theta}}{\partial x_j}\,\mathrm{d}V - \int_{\Gamma\sigma} \mathbf{N}_i (\bar{\sigma}_{ni} + \gamma p^n v_i)\,\mathrm{d}\Gamma = 0 \tag{28}$$

$$\text{(B)}\quad -\frac{\rho}{\Delta t}\int_V \frac{\partial \mathbf{N}}{\partial x_i}\,u_i^*\,\mathrm{d}V - \int_V \frac{\partial \mathbf{N}}{\partial x_i}\,\frac{\partial (p^{n+1} - \gamma p^n)}{\partial x_i}\mathrm{d}V + \frac{\rho}{\Delta t}\int_{\Gamma_u} \mathbf{N}\bar{u}_n^{n+1}\,\mathrm{d}\Gamma = 0 \tag{29}$$

$$\text{(C)}\quad \int_V \mathbf{N}_i \left\{ (u_i^{n+1} - u_i^*)\frac{\rho}{\Delta t} + \frac{\partial}{\partial x_i}(p^{n+1} - \gamma p^n) \right\} \mathrm{d}V - \int_{\Gamma_\sigma} \mathbf{N}_i (p^{n+1} - \gamma p^n)\mathrm{d}\Gamma = 0 \tag{30}$$

It must be noted that the essential and natural boundary conditions of Equations (29) are

$$p = 0 \quad \text{on } \Gamma_\sigma \tag{31}$$

$$\bar{u}^{n+1} \cdot v = 0 \quad \text{on } \Gamma_u \tag{32}$$

3.5.1. Discrete equations. Using approximations (22)–(24) the discrete equations become

$$\text{(A)}\quad \int_V \mathbf{N}_i \mathbf{N}_i^{\mathrm{T}}\,\mathrm{d}V \mathbf{U}_i^* = \int_V \mathbf{N}_i \mathbf{N}_i^{\mathrm{T}}\,\mathrm{d}V \mathbf{U}_i^n + \Delta t \int_V \mathbf{N}_i f_i\,\mathrm{d}V - \frac{\gamma \Delta t}{\rho}\int_V \mathbf{N}_i \frac{\partial \mathbf{N}^{\mathrm{T}}}{\partial x_i}\,\mathrm{d}V \mathbf{P}^n - \frac{\Delta t \mu}{\rho}\int_V \frac{\partial \mathbf{N}_i}{\partial x_j}\,\frac{\partial \mathbf{N}_i^{\mathrm{T}}}{\partial x_j}\,\mathrm{d}V \mathbf{U}_i^{n+\theta} + \frac{\Delta t}{\rho}\int_{\Gamma_\sigma} \mathbf{N}_i (\overline{\sigma}_{ni} + \gamma p^n)\,\mathrm{d}\Gamma \tag{33}$$

In compact form

$$\mathbf{M}\mathbf{U}^* = \mathbf{M}\mathbf{U}^n + \Delta t \mathbf{F} - \frac{\gamma \Delta t}{\rho}\mathbf{B}^{\mathrm{T}}\mathbf{P}^n - \frac{\Delta t \mu}{\rho}\mathbf{K}\mathbf{U}^{n+\theta}$$

and making use of the approximation described before for $\mathbf{U}^{n+\vartheta}$

$$\left(\mathbf{M}+\frac{\Delta t\mu\theta}{\rho}\mathbf{K}\right)\mathbf{U}^{*}=\mathbf{M}\mathbf{U}^{n}+\Delta t\mathbf{F}-\frac{\gamma\Delta t}{\rho}\mathbf{B}^{\mathrm{T}}\mathbf{P}^{n}-\frac{\Delta t\mu(1-\theta)}{\rho}\mathbf{K}\mathbf{U}^{n}$$

and for $\theta=1$ and $\gamma=0$

$$\text{(A)}\quad\left(\mathbf{M}+\frac{\Delta t\mu}{\rho}\mathbf{K}\right)\mathbf{U}^{*}=\mathbf{M}\mathbf{U}^{n}+\Delta t\mathbf{F} \tag{34}$$

In the same way

$$-\frac{\rho}{\Delta t}\int_{V}\left(\frac{\partial\mathbf{N}}{\partial x_i}\mathbf{N}_i^{\mathrm{T}}\right)\mathrm{d}V\mathbf{U}^{*}+\frac{\rho}{\Delta t}\int_{\Gamma_u}\mathbf{N}\bar{u}_n^{n+1}\,\mathrm{d}\Gamma=-\int_{V}\left(\frac{\partial\mathbf{N}}{\partial x_i}\frac{\partial\mathbf{N}^{\mathrm{T}}}{\partial x_i}\right)\mathrm{d}V(\mathbf{P}^{n+1}-\gamma\mathbf{P}^{n}) \tag{35}$$

In compact form

$$\mathbf{S}\mathbf{P}^{n+1}=\frac{\rho}{\Delta t}(\mathbf{B}\mathbf{U}^{*}-\hat{\mathbf{U}})+\mathbf{S}\gamma\mathbf{P}^{n}$$

and for $\theta=1$ and $\gamma=0$

$$\text{(B)}\quad\mathbf{S}\mathbf{P}^{n+1}=\frac{\rho}{\Delta t}(\mathbf{B}\mathbf{U}^{*}-\hat{\mathbf{U}}) \tag{36}$$

Finally

$$\int_{V}\mathbf{N}_i\mathbf{N}_i^{\mathrm{T}}\,\mathrm{d}V\mathbf{U}^{n+1}=\int_{V}\mathbf{N}_i\mathbf{N}_i^{\mathrm{T}}\,\mathrm{d}V\mathbf{U}^{*}-\frac{\Delta t}{\rho}\int_{V}\mathbf{N}_i\frac{\partial\mathbf{N}^{\mathrm{T}}}{\partial x_i}\,\mathrm{d}V(\mathbf{P}^{n+1}-\gamma\mathbf{P}^{n})$$
$$+\int_{\Gamma_\sigma}\mathbf{N}_i\mathbf{N}^{\mathrm{T}}\,\mathrm{d}\Gamma(\mathbf{P}^{n+1}-\gamma\mathbf{P}^{n}) \tag{37}$$

In compact form

$$\mathbf{M}\mathbf{U}^{n+1}=\mathbf{M}\mathbf{U}^{*}-\frac{\Delta t}{\rho}\mathbf{B}^{\mathrm{T}}(\mathbf{P}^{n+1}-\gamma\mathbf{P}^{n})$$

and for $\theta=1$ and $\gamma=0$

$$\text{(C)}\quad\mathbf{M}\mathbf{U}^{n+1}=\mathbf{M}\mathbf{U}^{*}-\frac{\Delta t}{\rho}\mathbf{B}^{\mathrm{T}}\mathbf{P}^{n+1} \tag{38}$$

where the matrices are

$$\mathbf{M}=\begin{bmatrix}\mathbf{M}_p & 0 & 0\\ 0 & \mathbf{M}_p & 0\\ 0 & 0 & \mathbf{M}_p\end{bmatrix} \tag{39}$$

$$\mathbf{M}_p = \int_V \mathbf{N}\mathbf{N}^{\mathrm{T}}\,\mathrm{d}V \tag{40}$$

$$\mathbf{B} = \left[\int_V \left(\frac{\partial \mathbf{N}}{\partial x}\mathbf{N}^{\mathrm{T}}\right)\mathrm{d}V; \int_V \left(\frac{\partial \mathbf{N}}{\partial y}\mathbf{N}^{\mathrm{T}}\right)\mathrm{d}V; \int_V \left(\frac{\partial \mathbf{N}}{\partial z}\mathbf{N}^{\mathrm{T}}\right)\mathrm{d}V\right] \tag{41}$$

$$\mathbf{S} = \int_V \left(\frac{\partial \mathbf{N}}{\partial x}\frac{\partial \mathbf{N}^{\mathrm{T}}}{\partial x} + \frac{\partial \mathbf{N}}{\partial y}\frac{\partial \mathbf{N}^{\mathrm{T}}}{\partial y} + \frac{\partial \mathbf{N}}{\partial z}\frac{\partial \mathbf{N}^{\mathrm{T}}}{\partial z}\right)\mathrm{d}V \tag{42}$$

$$\hat{\mathbf{U}} = \int_{\Gamma_u} \mathbf{N}\bar{u}_n^{n+1}\,\mathrm{d}\Gamma \tag{43}$$

$$\mathbf{K} = \begin{bmatrix} \mathbf{S} & 0 & 0 \\ 0 & \mathbf{S} & 0 \\ 0 & 0 & \mathbf{S} \end{bmatrix} \tag{44}$$

$$\mathbf{F}^{\mathrm{T}} = \left[\int_V \mathbf{N}^{\mathrm{T}} f_x\,\mathrm{d}V; \quad \int_V \mathbf{N}^{\mathrm{T}} f_y\,\mathrm{d}V; \quad \int_V \mathbf{N}^{\mathrm{T}} f_z\,\mathrm{d}V\right] + \frac{1}{\rho}\left[\int_{\Gamma_\sigma} \mathbf{N}^{\mathrm{T}}\bar{\sigma}_{nx}\,\mathrm{d}\Gamma; \quad \int_{\Gamma_\sigma} N^{\mathrm{T}}\bar{\sigma}_{ny}\,\mathrm{d}\Gamma; \quad \int_{\Gamma_\sigma} \mathbf{N}^{\mathrm{T}}\bar{\sigma}_{nz}\,\mathrm{d}\Gamma\right] \tag{45}$$

3.6. Summary of a full iterative time step

A full time step may be described as follows: starting with the known values u^n and p^n in each particle, the computation of the new particle position involves the following steps:

(I) Approximate u^{n+1} (For the first iteration $u^{n+1} = 0$. For the subsequent iterations the value of u^{n+1} corresponding to the last iteration is taken).
(II) Move the particles to the x^{n+1} position and generate a mesh.
(III) Evaluate the u^* velocity from (34). (It must be noted that the matrices M and K are separated in 3 blocks. Then, this equations may be solved separately for U_x^*, U_y^* and U_z^*. For $\theta \neq 0$ (implicit) involves the solution of 3 Laplacian equations. For $\theta = 0$ (explicit) the M matrix may be lumped and inverted directly).
(IV) Evaluate the pressure p^{n+1} by solving the Laplacian Equation (36).
(V) Evaluate the velocity u^{n+1} using (38). Go to (I) until convergence.

The Lagrangian split scheme described has two important advantages:

(1) Step III is linear and may be explicit ($\theta = 0$) or implicit ($\theta \neq 0$). The use of a Lagrangian formulation eliminates the standard convection terms present in Eulerian formulations. The convection terms are responsible for non-linearity, non-symmetry and non-self-adjoint operators which require the introduction of high-order stabilization terms to avoid numerical

oscillations. All these problems are not present in this formulation. Only the non-linearity remains due to the unknown of the final particle position.

(2) In all the steps, the system of equations to be solved are the evaluation of the velocity components (step III) and the evaluation of the pressure (step IV). Those systems are scalar (only one degree of freedom by node), symmetric and positive definite. Then, it is very easy to solve them using a symmetric iterative scheme (such as the conjugate gradient method).

3.7. Stabilization of the incompressibility condition

In the Eulerian form of the momentum equations, the discrete form must be stabilized in order to avoid numerical wiggles in the velocity and pressure results. This is not the case in the Lagrangian formulation where no stabilization parameter must be added in Equations (34) and (38). Nevertheless, the incompressibility condition must be stabilized in equal-order approximations to avoid possible pressure oscillations in some particular cases.

For instance for small pressure split ($\gamma \neq 0$) or for small time step increments (Courant number much less than one) it is well known that the fractional step does not stabilize the pressure waves. In those particular cases, a stabilization term must be introduced in Equations (B) in order to eliminate pressure oscillations.

A simple and effective procedure to derive a stabilized formulation for incompressible flows is based on the so-called Finite Calculus formulations [19–21].

In all the examples presented in this paper, the γ parameter was always fixed equal to zero and the time increments were fixed to a given value of the Courant number ≈ 1, avoiding in this way all the stabilization problems.

4. BOUNDARY SURFACES RECOGNITION

One of the main problems in mesh generation is the correct definition of the boundary domain. Sometimes, boundary nodes are explicitly defined as special nodes, which are different from internal nodes. In other cases, the total set of nodes is the only information available and the algorithm must recognize the boundary nodes. Such is the case in Particle Methods in which, at each time step, a new particle position is obtained and the boundary-surface must be recognized using the new particle positions.

The use of the MFEM with the extended Delaunay partition makes it easier to recognize boundary nodes.

Considering that the particles follow a variable $h(x)$ distribution, where $h(x)$ is the minimum distance between two particles, the following criterion has been used:

All particles on an empty sphere with a radius $r(x)$ bigger than $\alpha h(x)$ are considered as boundary particles (see Figure 2).

Thus, α is a parameter close to, but greater than one. Note that this criterion is coincident with the Alpha Shape concept [22].

Once a decision has been made concerning which of the particles are on the boundaries, the boundary surface must be defined. It is well known that in 3D problems the surface fitting a number of particles is not unique. For instance, four boundary particles on the same sphere may define two different boundary surfaces, a concave one and convex one.

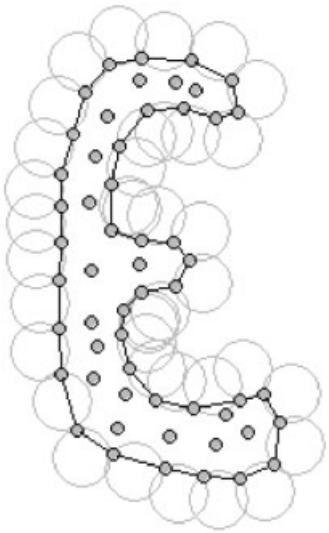

Figure 2. Contour recognition: Empty circles with radius $\alpha h(x)$ define the boundary particles.

In this work, the boundary surface is defined with all the polyhedral surfaces having all their particles on the boundary and belonging to just one polyhedron. See Reference [13].

The correct boundary surface may be important to define the correct normal external to the surface. Furthermore, in weak forms (Galerkin) a correct evaluation of the volume domain is also important. Nevertheless, it must be noted that in the criterion proposed above, the error in the boundary surface definition is proportional to h. This is the error order accepted in a numerical method for a given node distribution. The only way to obtain more accurate boundary surface definition is by decreasing the distance between the particles.

5. NUMERICAL RESULTS

A number of free-surface flow and fluid–structure interaction problems will be presented. In a first group of examples the interacting solid will be considered infinitely rigid and fixed. Those cases are useful to compare the results with experimental and analytical ones. The interacting solid will also be represented with particles but with imposed velocity equal to zero. In a second group of examples moving rigid solid motions will be considered. In all cases, the elastic strains will be neglected. The solid will be considered in two different ways:

(a) As a particular material with a high viscosity parameter, much higher than the fluid domain. For practical purposes a $10^{10}\mu$ value will be considered. This value is enough to represent a solid without introducing numerical problems.
(b) The solid will be considered as a boundary contour with an imposed velocity. After each time step, the fluid forces on the solid due to the pressure and the viscous terms will be evaluated. In the next step, the solid will move rigidly using Newton law.

Time stepping and iterative process: The time step length Δt was imposed to a variable value and evaluated at the beginning of each time step. The criterion to calculate the time step

Table I. Average time (in s) for a standard PC considering 3 iterations by time step.

n	Monolithic	Fractional step	Mesh generation
10^4	69 s	30 s	7 s
10^5	46 min	20 min	1.5 min
10^6	1829 min	796 min	18 min

was: during the iterative implicit process (Section 6.3), the time step may be as big as possible with the limitation that at the end of the iteration any element cannot have a negative or zero volume. In this way the mesh is preserved during the entire time step. This criterion is less restrictive than imposing a Courant number less than one.

In all the examples performed, a maximum of 3 iterations in the iterative process (see table on Section 3.6) was needed to reach a reasonable convergence.

Computing time: The computing time of each time step is of the same order as a standard incompressible fluid mechanics problem solved via a fractional step method, adding the computing time needed to generate the polyhedral mesh and the boundary recognition. One of the key points for the success of the Lagrangian flow formulation described here is the fast generation of a mesh. In Reference [14] it is shown that the EDT and the Alpha-Shape method solve this problem in order $n^{1.1}$. In particular, Reference [14] presents the computing time evaluations for different size problems performed in a standard PC of 1 GHz. The computing time in second for the mesh generation and boundary recognition is

$$t(\mathrm{s}) = 0.00283n^{1.1}$$

This time must be compared with the computing time needed to solve a Laplacian equation. This is very problem dependent, but for a standard 3D problem using a conjugate gradient iterative method, an optimist number of operation to achieve a reasonable convergence error is of order $n^{1.6}$. For the monolithic case, (the entire unknown solved together) this means to solve a system of 4n degrees of freedom (d.o.f). Supposing 3 iterations by time step, this means for a 1 GHz PC:

$$t(\mathrm{s}) = 10^{-6}3(4n)^{1.6}$$

For the fractional step method, this means to solve 4 Laplacian of n d.o.f. at each iteration. Considering also 3 iterations by time step in the same PC means

$$t(\mathrm{s}) = 10^{-6}3^{*}4n^{1.6}$$

Table I shows a comparison of the computing time for different number of particles. It is clear that for large d.o.f problems, the computing time needed to evaluate a new mesh at each time step is not important compared with computing time involved to solve the non-linear system.

5.1. Sloshing problems

The simple problem of the free oscillation of an incompressible liquid in a container is considered first. Numerical solutions for this problem can be found in several references [23].

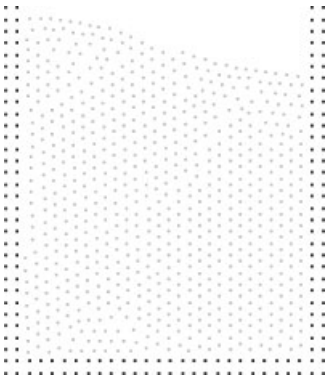

Figure 3. Sloshing. Initial point distribution.

This problem is interesting because there is an analytical solution for small amplitudes. Figure 3 shows a schematic view of the problem, and the point distribution in the initial position. The dark points represent the fixed points where the velocity is fixed to zero. It is worth mentioning that in this problem the wall has been represented by two layers of nodes but the elements constructed between layers are omitted from the integration process. Thus, the nodes on the external layer do not take part in the computation and are included in the figure only for visualization purposes.

Figure 4 shows the variation in time of the amplitude compared with the analytical results for the near inviscid case. Little numerical viscosity is observed on the phase wave and amplitude in spite of the relative poor point distribution.

The analytical solution is only acceptable for small wave amplitudes. For larger amplitudes, additional waves are overlapping and finally, the wave breaks and also some particles can be separated from the fluid domain due to their large velocity. Figure 5 shows the numerical results obtained with the method presented in this paper for larger sloshing amplitudes. Breaking waves as well as separation effects can be seen on the free-surface. This particular and very complicated effect is apparently well represented by this model.

In order to test the potentiality of the method in a 3D domain, the same sloshing problem was solved as a 3D problem. Figure 6 shows the different point position at two time steps. Each point position was represented by a sphere and only a half of the fixed recipient is represented on the figure. The sphere representation is used only to improve the visualization of the fluid movement.

5.2. Dam collapse

This problem was solved by Koshizuka and Oka [4] both experimentally and numerically in a 2D domain. It became a classical example to test the validation of the Lagrangian formulation in fluid flows. In this paper, the results obtained using the method proposed in 2D and 3D domains are presented. The water is initially located on the left supported by a removable board. See Figure 7. The collapse starts at time `t=0`, when the removable board is slid-up. Viscosity and surface tension are neglected.

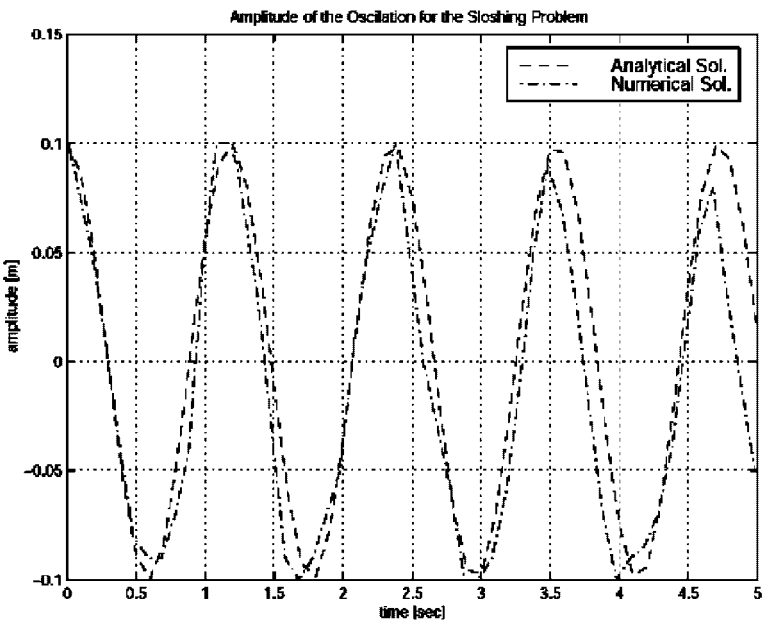

Figure 4. Sloshing: Comparison of the numerical and analytical solution.

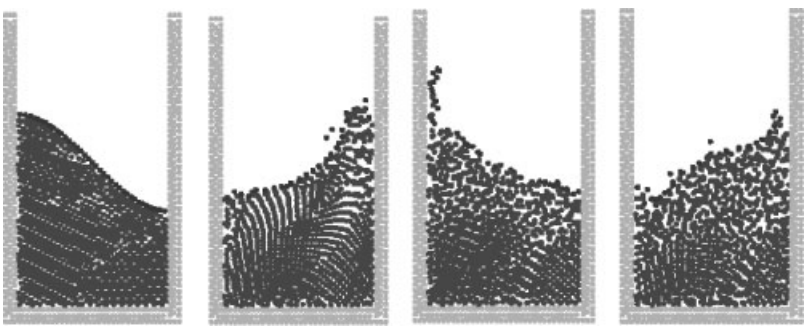

Figure 5. Sloshing: Different time step for large amplitudes.

Figures 8(a)–(d) show the point positions at different time steps. The blue points represent the free-surface detected with the alpha-shape algorithm with an alpha parameter $\alpha = 1.1$. The internal points are sky-blue and the fixed wall is yellow in the 3D and brown in the 2D example.

The water is running on the bottom wall until, near 0.3 s, it impinges on the right vertical wall. Breaking waves appear at 0.6 s. Around $t = 1$ s the main water wave reaches the left wall again Agreement with the experimental results of Reference [4] both in the shape of the free surface and time development is excellent.

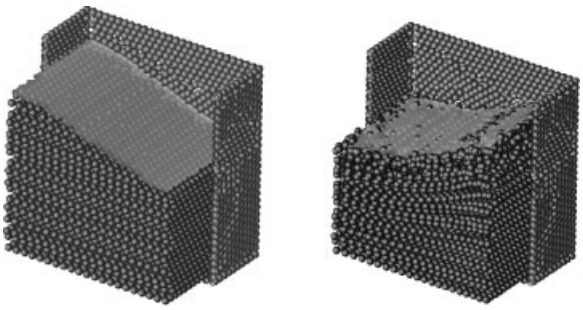

Figure 6. Sloshing: Different time step for 3D domains.

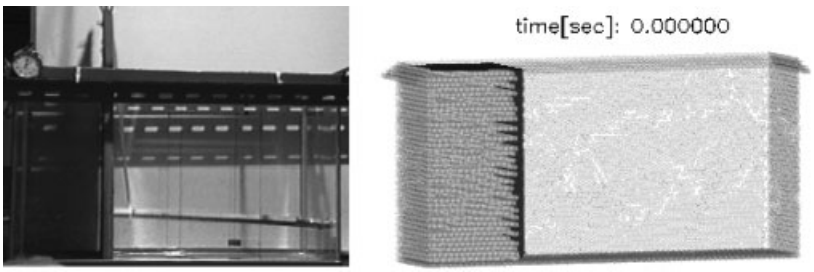

Figure 7. Dam Collapse. Initial position. Left: experimental [6]. Right: 3D simulation.

In this example, the power of the method to represent breaking waves and flow separation for a very complicated and random problem is verified and compared with experimental results.

5.3. Wave breaking on a beach

A simulation of the propagation of a water wave and its breaking due to shoaling over a plane slope is presented next. This example was numerically studied in Reference [23] with a Lagrangian formulation using directly the standard Finite Element Method with remeshing. There is also an analytical solution for a simplified approximation that is used for comparison [24]. Figure 9 shows the initial point distribution and Figure 10(a) comparison with the analytical free-surface at a different time step. The geometry of the problem as well as a discussion of the analytical solution may be found in Reference [23].

Initially (Figures 10(a) and (b) the wave travels over a constant depth bottom towards the slope with no ostensible change of shape. Strongly non-linear effects appear when the wave hits the slope (Figure 10(c)). The crest of the wave accelerates while the rest lags behind (Figure 10(d)). At this time the comparisons with the analytical solution are in agreement only in the wave position. The shape of the wave obtained with the numerical solution is totally different. The reason is that the analytical solution gives symmetrical shape waves, which are not physical, before the breaking process. Subsequently, a water jet is formed at the crest

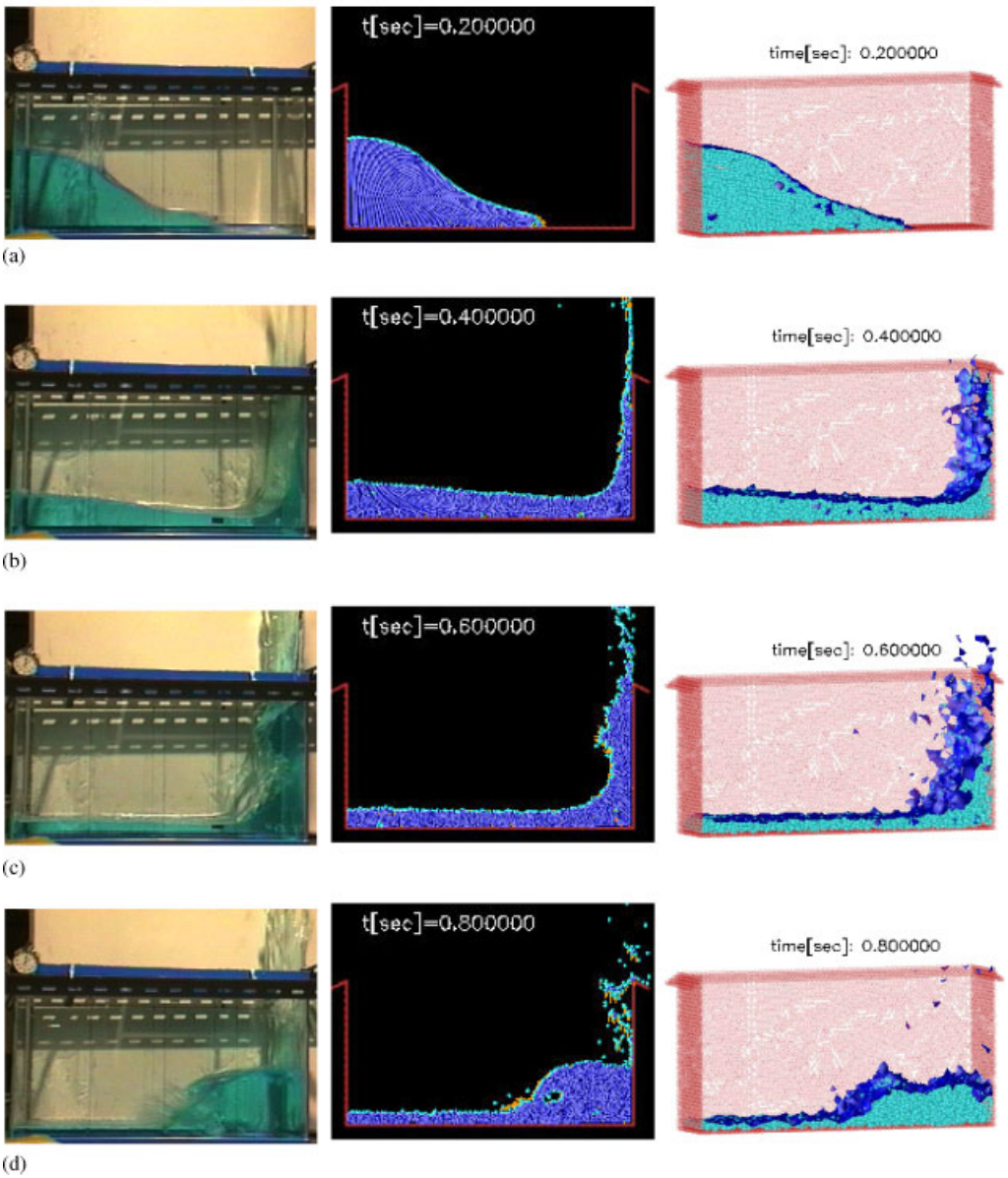

Figure 8. Dam Collapse. Comparison with experimental results of Reference [6]: (a) experimental, 2D and 3D numerical solution at $t = 0.2$ s; (b) experimental, 2D and 3D numerical solution at $t = 0.4$ s; (c) experimental, 2D and 3D numerical solution at $t = 0.6$ s; and (d) experimental, 2D and 3D numerical solution at $t = 0.8$ s.

plunge making the breaking wave (Figures 10(e) and (f)) and coming in contact with the nearly still surface of the water ahead. In Reference [23] the evaluation is stopped before this contact point. Using the methodology proposed in this paper, the analysis may be continued

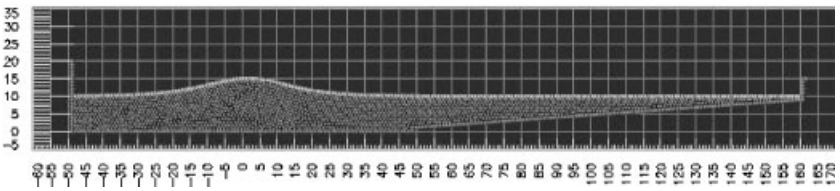

Figure 9. Wave breaking on a beach. Initial geometry and point position.

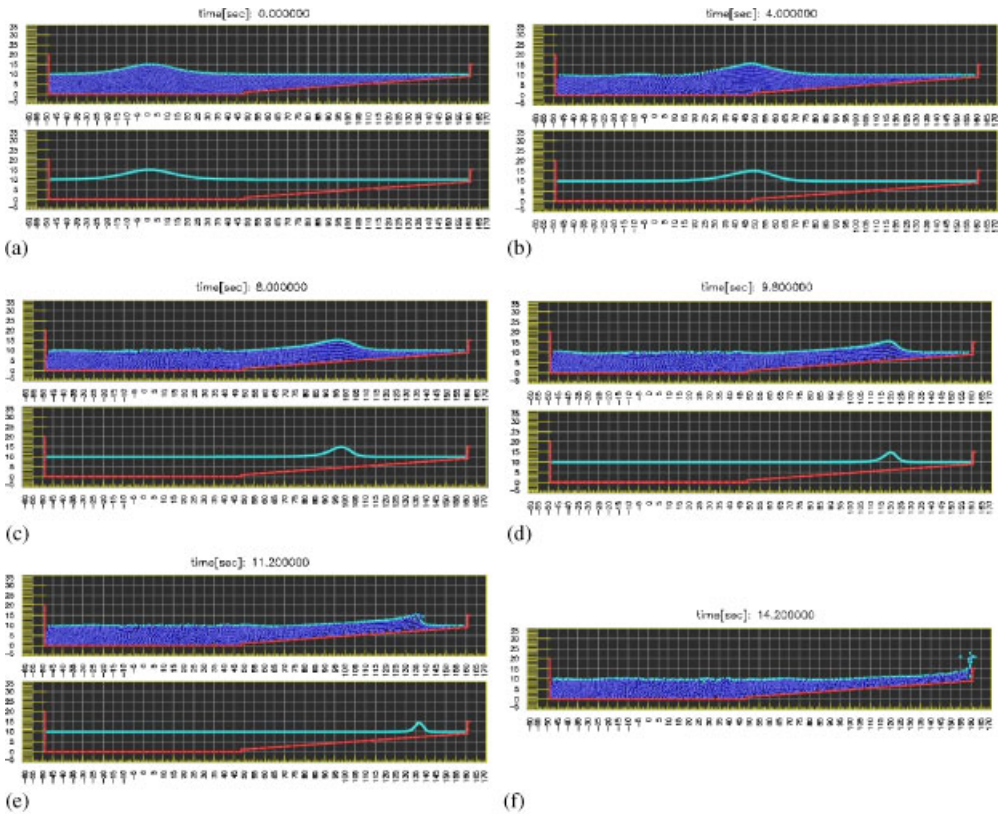

Figure 10. Wave breaking on a beach. Comparison with analytical results at different time steps. Top: Numerical solution. Bottom: Analytical solution: (a) $t = 0$ sec; (b) $t = 4$ sec; (c) $t = 8$ s; (d) $t = 9.8$ s; and (e) $t = 11.2$ sec, (f) $t = 14.2$ s.

until the end. In Figures 10(g) and (h), the wave finally hits a lateral wall (introduced in the model to stop the lateral effects) producing drop separations, and then coming back towards the left as a new wave.

The ability of the model to accurately simulate the various stages of the wave breaking is noteworthy.

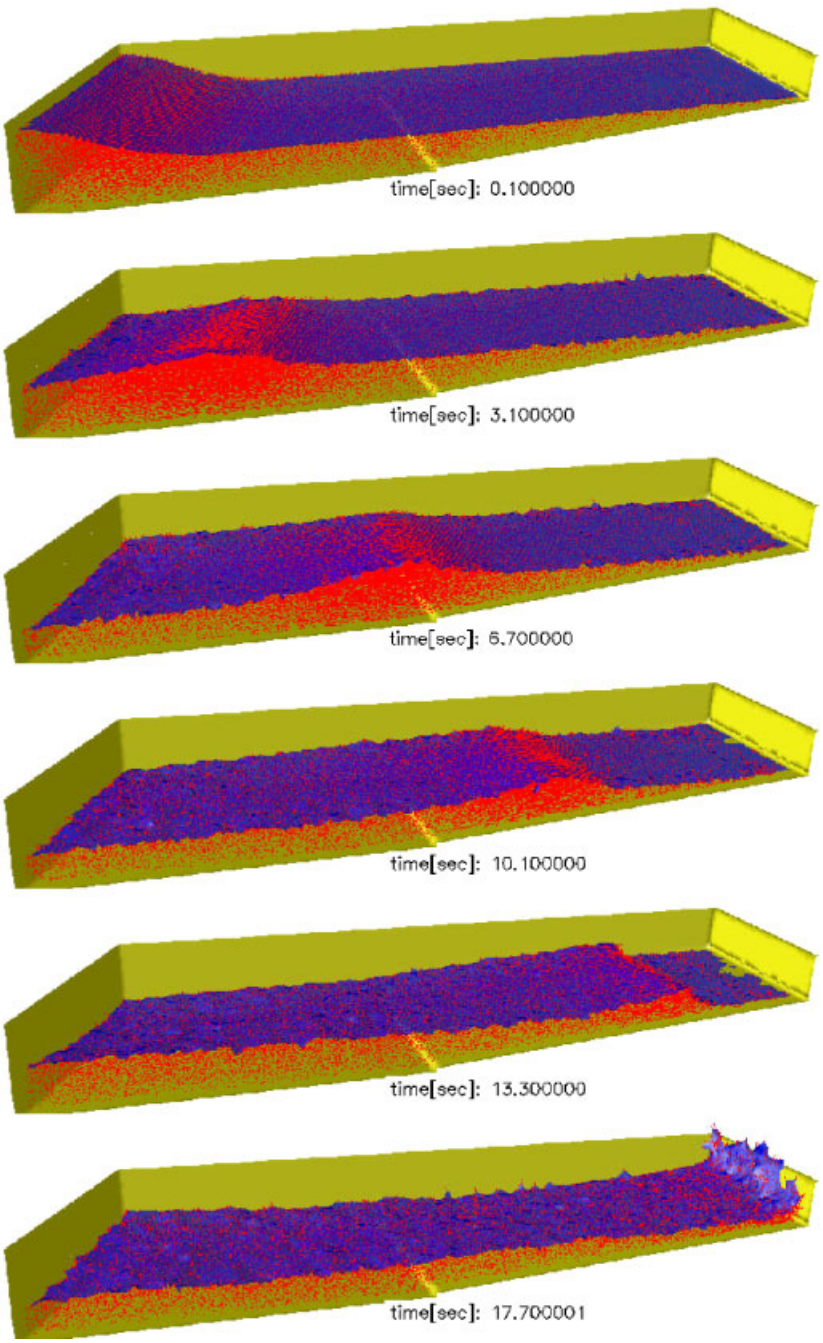

Figure 11. Breaking wave on a beach: Oblique wave on a 3D domain.

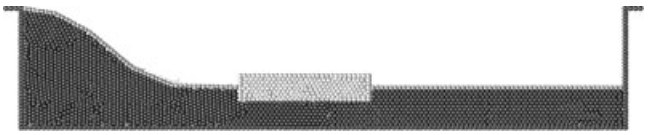

Figure 12. Solid floating on a free-surface. Initial geometry and point distribution.

Nevertheless, a 2D domain is an easy case and may be solved acceptably with any mesh generator. The true problems appear in a 3D domain, where the mesh generation is complicated with the presence of slivers and other geometric mesh generation problems. In order to show the power of the tool presented, the same problem was solved in a 3D domain.

To transform the wave breaking described before in a true 3D problem, the initial position of the wave was introduced having an oblique angle with the beach line. In this way, a 3D effect appears. When the wave hits the slope, the crest of the wave accelerates differently in accordance with the depth, inducing the wave to correct its oblique position and break parallel to the beach. The results may be seen in Figure 11 for different time steps.

5.4. Solid floating on a free-surface

The following example, shown schematically in Figure 12, represents a very interesting problem of fluid–structure interaction when there is a weak interaction between the fluid and a large rigid deformation of the structure. In this case, there is also a free-surface problem, representing a schematic case of sea-keeping in ship hydrodynamics.

The example shows a recipient with a floating piece of wood in which a wave is produced on the left side. The wave intercepts the wood piece producing a breaking wave and moving the floating wood. In this example the solid was represented by very viscous flows with a viscosity parameter order ten times greater than the water viscosity. Figure 13 shows the pressure contours and the free-surface position for different time steps.

This example, as well as the next example to be presented in Section 5.5, has no analytical or experimental result to use as comparison. The reason to present it in this paper is to show the possibility of the method to carry out fluid–structure interaction problems. The behaviour of the solid seems to be correct and the flow moving is acceptably realistic.

5.5. Solid cube falling in a recipient with water

This last example is also a case of fluid–structure interaction. The solid is initially totally free and is falling down into a recipient with a fluid. Figure 14 shows the initial position and the initial mesh. In this example, the solid was modelled as a boundary condition for the fluid. Once the pressure and the viscous forces have been evaluated in the fluid, the solid is accelerated using Newton law. The solid has a mass and a gravity force concentrate in its gravity centre. The solid is considered to be light compared to the liquid weight.

At the beginning the solid falls free due to the gravity forces. Once in contact with the water free-surface ($t = 0.31$ s) the alpha-shape method recognizes the different boundary contours.

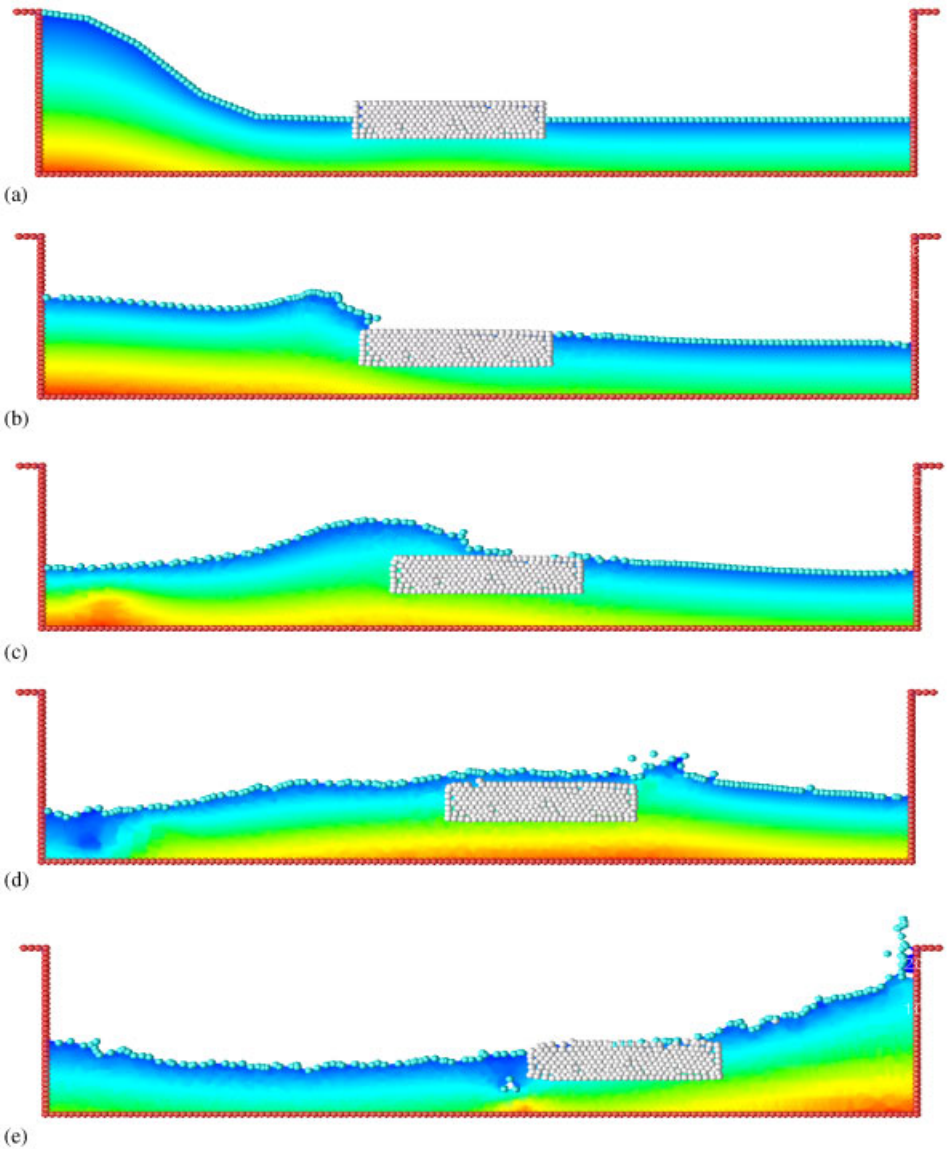

Figure 13. Solid floating on a free-surface. Pressure contours and free-surface positions for different time steps: (a) $t = 0$ s; (b) $t = 0.29$ s; (c) $t = 0.49$ s; (d) $t = 0.71$ s; and (e) $t = 1.23$ s.

For instance, the red points on the solid cube are dry particles while the blue points on the solid cube are wet particles. The sky-blue points are free-surface points.

The pressure and the viscous forces are evaluated in the entire domain and in particular on the solid cube. This flow forces introduce a negative acceleration to the vertical velocity until, once the solid is completely inside the water, the falling velocity becomes zero. Then,

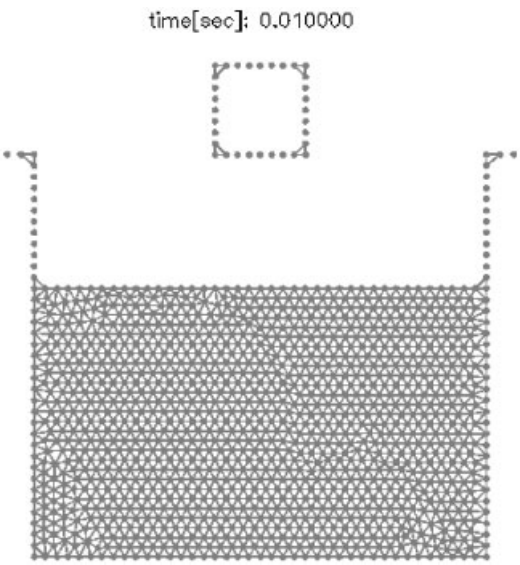

Figure 14. Solid cube falling into a recipient with water. Initial mesh.

Arquimides principle makes the solid to go up to the free-surface. Figure 15 shows different time steps. It is interesting to observe that there is a rotation of the solid. This is due to the fact that the centre of the floating forces is higher in the rotated position than in the initial ones.

6. CONCLUSIONS

Particle Methods combined with a Finite Element Method in which the meshes are generated linearly with the number of particles are an excellent tool to solve fluid mechanic problems, especially fluid–structure interactions with moving free-surfaces.

The Meshless Finite Element Method seems to be the best adapted FEM to this kind of combination. In fact, the MFEM has the advantages of a meshless method concerning the easy introduction of the nodes connectivity in a bounded time of order n. The method also preserve the classical advantages of the FEM such as: (a) the simplicity of the shape functions, (b) C_0 continuity between elements, (c) an easy introduction of the boundary conditions, and (d) symmetric matrices.

The fractional step approach presented here has proved to be an efficient procedure for solving accurately the Lagrangian flow equations.

Both Particle Methods and the MFEM are the key ingredients to the Particle Finite Element Method, a very suitable method to solve fluid–structure interaction problems including free-surface, breaking waves, flow separations, contact problems and collapse situations.

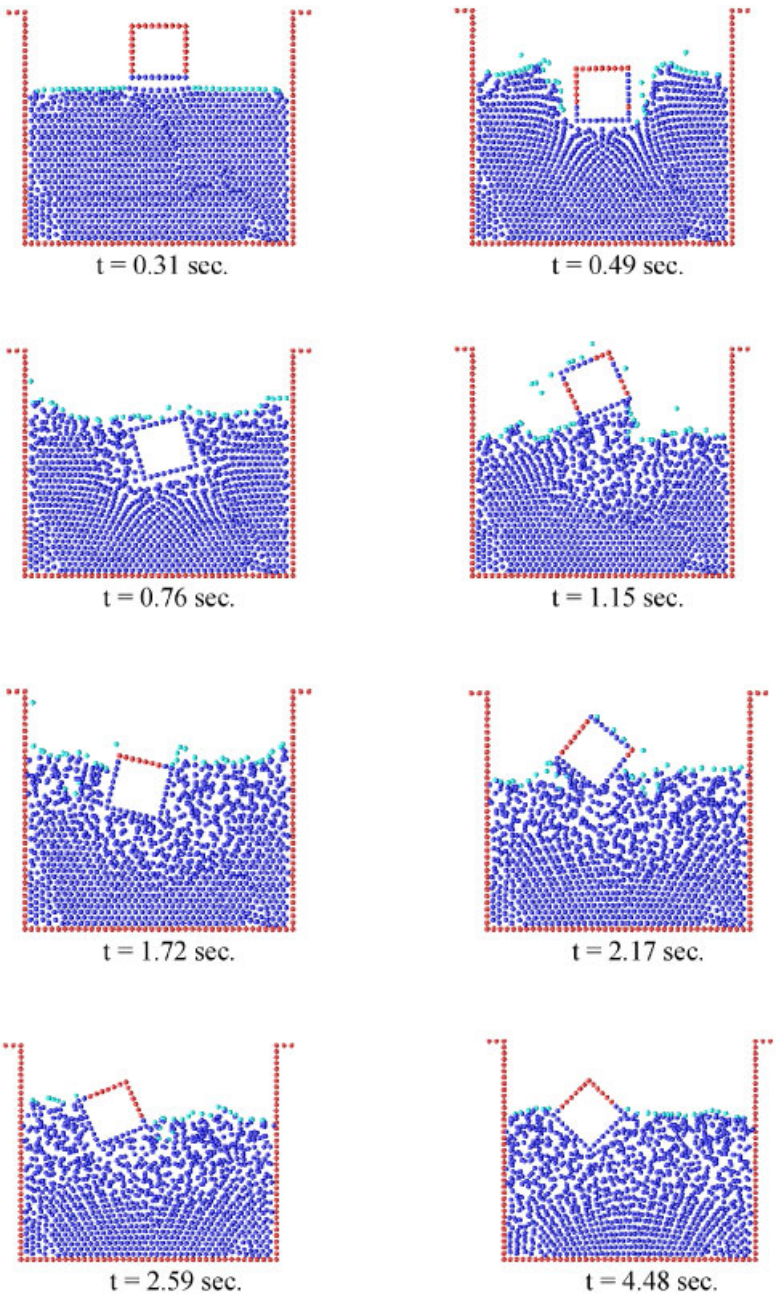

Figure 15. Solid cube falling into a recipient with water. Different time steps.

REFERENCES

1. Gingold RA, Monaghan JJ. Smoothed particle hydrodynamics, theory and application to non-spherical stars. *Monthly Notices of the Royal Astronomical Society* 1997; **181**:375–389.
2. Bonet J, Kulasegaram S. Correction and stabilization of smooth particle hydrodynamics methods with applications in metal forming simulation. *International Journal for Numerical Methods in Engineering* 2000;**47**:1189–1214.
3. Dilts GA. Moving least squares particle hydrodynamics. I. Consistency and stability. *International Journal for Numerical Methods in Engineering* 1999; **44**:1115–1155.
4. Koshizuka S, Oka Y. Moving particle semi-implicit method for fragmentation of incompressible fluid. *Nuclear Engineering Science* 1996; **123**:421–434.
5. Crowley WP. *Lecture Notes in Physics*. Springer: Berlin, 1970; 8–37.
6. Fritts MJ, Crowley WP, Trease HE. *The Free Lagrange Method*, Lecture Notes in Physics, vol. 238. Springer: New York, 1985.
7. Belytschko T, Liu Y, Gu L. Element free Galerkin methods. *International Journal for Numerical Methods in Engineering* 1994; **37**:229–256.
8. De S, Bathe KJ. The method of finite spheres with improved numerical integration. *Computers and Structures* 2001; **79**:2183–2196.
9. Nayroles B, Touzot G, Villon P. Generalizing the fem: diffuse approximation and diffuse elements. *Computational Mechanics* 1992; **10**:307–318.
10. Oñate E, Idelsohn SR, Zienkiewicz OC, Taylor RL. A finite point method in computational mechanics. Applications to convective transport and fluid flow. *International Journal for Numerical Methods in Engineering* 1996; **39**(22):3839–3886.
11. Oñate E, Idelsohn SR, Zienkiewicz OC, Taylor RL, Sacco C. A stabilized finite point method for analysis of fluid mechanics problems. *Computer Methods in Applied Mechanics and Engineering* 1996; **39**:315–346.
12. Idelsohn SR, Storti MA, Oñate E. Lagrangian formulations to solve free surface incompressible inviscid fluid flows. *Computer Methods in Applied Mechanics and Engineering* 2001; **191**:583–593.
13. Idelsohn SR, Oñate E, Calvo N, Del Pin F. The meshless finite element method. *International Journal for Numerical Methods in Engineering* 2003; **58**(6):893–912.
14. Calvo N, Idelsohn SR, Oñate E. Polyhedrization of an arbitrary 3D point set. *Computer Method in Applied Mechanics and Engineering* 2003; **192**:2649–2667.
15. Rojek J, Oñate E, Zarate F, Miquel J. Modelling of rock, soil and granular materials using spherical elements. *Proceedings of the European Conference on Computer Mechanics* (*ECCM 2001*), Cracow, Poland, June 2001.
16. Idelsohn SR, Oñate E, Del Pin F. A Lagrangian meshless finite element method applied to fluid–structure interaction problems. *Computer and Structures* 2003; **81**:655–671.
17. Codina R. Pressure stability in fractional step finite element methods for incompressible flows. *Journal of Computational Physics* 2001; **170**:112–140.
18. Belikov V, Semenov A. Non-sibsonian interpolation on arbitrary system of points in Euclidean space and adaptive generating isolines algorithm. *Numerical Grid Generation in Computational Field Simulation*, *Proceedings of the 6th International Conference*. *Greenwich University*, July 1998.
19. Oñate E. Derivation of stabilized equations for advective–diffusive transport and fluid flow problems. *Computer Methods in Applied Mechanics and Engineering* 1998; **151**(1–2):233–267.
20. Oñate E. A stabilized finite element method for incompressible viscous flows using a finite increment calculus formulation. *Computer Methods in Applied Mechanics and Engineering* 2002; **182**(1–2):355–370.
21. Oñate E. Possibilities of finite calculus in computational mechanics. *International Journal for Numerical Methods in Engineering* 2004; **60**:255–281.
22. Edelsbrunner H, Mucke EP. Three-dimensional alpha-shape. *ACM Transactions on Graphics* 1994; **3**:43–72.
23. Radovitzky R, Ortiz M. Lagrangian finite element analysis of a Newtonian flows. *International Journal for Numerical Methods in Engineering* 1998; **43**:607–619.
24. Laitone EV. The second approximation to cnoidal waves. *Journal of Fluid Mechanics* 1960; **9**:430.

