Available online at www.sciencedirect.com

SCIENCE DIRECT®

Journal of Computational Physics 205 (2005) 611–625

JOURNAL OF COMPUTATIONAL PHYSICS

www.elsevier.com/locate/jcp

Meshless local Petrov–Galerkin method for two-dimensional nonlinear water wave problems

Q.W. Ma *

School of Engineering and Mathematical Science, City University, Northampton Square, London EC1V 0HB, UK

Received 24 May 2004; received in revised form 22 October 2004; accepted 17 November 2004
Available online 19 December 2004

Abstract

In this paper, the meshless local Petrov–Galerkin (MLPG) method is extended to dealing with nonlinear water wave problems. The formulation is based on general fluid governing equations and a time marching procedure. At each time step, the boundary value problem for the pressure is solved using the MLPG method; and the velocity and position of nodes are updated by numerical integration. The newly-extended method is applied to simulating water waves generated by a wave maker and good agreement with analytical solutions and finite element results is presented.

Keywords: Water waves; Meshless local Petrov–Galerkin method; Free surface

1. Introduction

Simulation of non-linear water waves has received numerous studies, for which mesh-based methods, such as finite element, finite volume and finite difference methods, are widely used. These methods have provided many useful and satisfactory results. However, their successes largely rely on good quality meshes. The construction of those meshes is usually a difficult and time-consuming task because it must be ensured that the aspect ratios of all elements are not very large and not very small and the connectivity between nodes and elements must be carefully and accurately found and recorded. In addition, elements can frequently become over-distorted during the simulation of water wave evolutions. The over-distorted elements may be amended by remeshing. However, remeshing can be as expensive as the generation of original meshes and may take a major proportion of computational costs considering that this has to be done

* Tel.: +44 20 7040 8159; fax: +44 20 7040 8566.
E-mail address: q.ma@city.ac.uk.

0021-9991/$ - see front matter © 2004 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2004.11.010

frequently or even every time step. Although the over-distortion problems may not arise if using fixed meshes, numerical diffusions due to advection terms may become severe and the motion of a floating body is not easy to cope with in such cases.

A new class of methods has recently been developed, which can overcome the problems associated with mesh-based methods. These methods do not need the use of any mesh to discretise computational domains and to construct approximate solutions. Alternatively, they are only based on randomly-ordered and -distributed nodes, implying that the problems associated with the shape and connectivity of elements in mesh-based methods vanish automatically in these meshless (or particle) methods. Many meshless (or particle) methods have been reported in the literature, such as element free Galerkin method [1], diffusion element method [2], reproducing kernel particle method [3], smooth particle hydrodynamics method [4], particle finite element method [5] and so on. Among them, the smoothed particle hydrodynamics (SPH) method has been most widely used to simulate water wave problems (see, e.g. [4,6,7]).

More recently, another meshless method, called meshless local Petrove–Galerkin (MLPG) method, has been developed in [8,9]. This method is based on a local weak form over local sub-domains (circles for two dimensional problems and spheres for three dimensional ones). It offers great flexibility. Many other meshless methods can be considered to be special cases of the MLPG method. The success of the MLPG method has been reported in solving fracture mechanics problems [10], beam and plate bending problems [11], three dimensional elasto-static and -dynamic problems [12,13] and some fluid dynamic problems such as steady flow around a cylinder [8], steady convection and diffusion flow [14] in one and two dimensions and lid-driven cavity flow in a two dimensional box [15]. Nevertheless, publications have not been found so far to use this method for solving flow problems with a free surface.

In this paper, the MLPG method will be extended to simulating nonlinear water waves. These problems are of great importance to offshore engineering but are difficult to deal with. They do not only have the features of general fluid dynamic problems but also have an unknown moving free surface. Any new numerical method, even successfully applied to other fluid problems, is usually required to be carefully refined when it is applied to water waves. It is the purpose of this paper to explore the potential of the new developed MLPG method in modelling the propagation of nonlinear water waves.

2. Governing equation and numerical procedure

The flow of incompressible and non-viscous fluids is considered, which is governed by the following equations and conditions:

$$\nabla \cdot \vec{u} = 0, \tag{1a}$$

$$\frac{\mathrm{D}\vec{u}}{\mathrm{D}t} = -\frac{1}{\rho}\nabla p + \vec{g}, \tag{1b}$$

$$\frac{\mathrm{D}\vec{x}}{\mathrm{D}t} = \vec{u} \quad \text{and} \quad p = p_{\mathrm{atm}} \text{ on the free surface,} \tag{2a}$$

$$\vec{u}.\vec{n} = \vec{U}.\vec{n} \quad \text{and} \quad \vec{n}.\nabla p = \rho\left(\vec{n}\cdot\vec{g} - \vec{n}\cdot\dot{\vec{U}}\right) \text{ on solid boundaries,} \tag{2b}$$

where ρ is the density of fluids, $\vec{u}$ the velocity of fluids, $\vec{U}$ the velocity of solid boundaries, $\vec{g}$ the gravitational acceleration, p the pressure and p_{atm} the atmospheric pressure.

It is well known that the non-viscous flow can be formulated by a velocity potential. However, because our research efforts will not be restricted to non-viscous flow in future, the potential formulation is not adopted here.

The above equations will be solved using a time-step marching procedure. This starts from a particular instant when the velocity and the geometry of fluid flow are known and then evolves to next time step, at which all physical quantities are updated by solving the governing equations. During each time step, the problem is formulated using a well known time-split procedure. This formulation contains three sub-steps:

(1) Evaluating intermediate velocities and positions

At the start of a new time step, the intermediate velocities and positions are first evaluated by

$$\vec{u}^{(*)} = \vec{u}^{(n)} + \vec{g}\Delta t, \tag{3}$$

$$\vec{r}^{(*)} = \vec{r}^{(n)} + \vec{u}^{(*)}\Delta t, \tag{4}$$

where $\vec{r}$ is the position vector of a point; Δt is the time step; and the superscripts (n) represent the quantities at time $t = t_n$. On this basis, the velocities at time $t = t_{n+1} = t_n + \Delta t$ can be expressed by

$$\vec{u}^{(n+1)} = \vec{u}^{(*)} + \vec{u}^{(**)}. \tag{5}$$

(2) Estimating the pressure

Integrating Eq. (1b) over the time interval (t_n, t_{n+1}) results in

$$\vec{u}^{(n+1)} = \vec{u}^{(n)} + \vec{g}\Delta t - \int_{t_n}^{t_{n+1}} \left(\frac{1}{\rho}\nabla p\right) \mathrm{d}t.$$

The integration with respect to time can be approximated by well-known methods, such as explicit, implicit or semi-implicit methods. The implicit method is used in this work, which gives

$$\vec{u}^{(n+1)} = \vec{u}^{(n)} + \vec{g}\Delta t - \frac{\Delta t}{\rho}\nabla p^{(n+1)}$$

and so

$$\vec{u}^{(**)} = -\frac{\Delta t}{\rho}\nabla p^{(n+1)}. \tag{6}$$

The velocity in Eq. (5) must also satisfy the continuity Eq. (1a), leading to

$$\nabla^2 p^{(n+1)} = \frac{\rho}{\Delta t}\nabla \cdot \vec{u}^{(*)}. \tag{7}$$

This is the equation governing the pressure at the new time level.

(3) Computing the velocity and the position at time $t = t_{n+1}$

After the solution for $p^{(n+1)}$ is found, the velocities can be estimated using Eq. (5) and the positions of fluids can be updated by numerically integrating the velocities.

The time split procedure was initially introduced by Chorin [16] for incompressible flow in 1968. It has been extended to deal with various problems using a finite element method as described in the book of Zienkiewicz and Taylor [17]. The procedure has also been employed by a number of authors who use other meshless or particle methods, e.g. [7,18,19].

Although the above formulation is employed for the non-viscous flow it may be extended to viscous flow by adding viscous stress terms to Eq. (3). In that case, however, iterations may be necessary since the viscous stresses are estimated using the velocities at previous steps, which would require to be improved for coping with complicated phenomena in viscous flows. In addition, this formulation can be directly applied to three dimensional problems, though only two dimensional ones are dealt with in this paper.

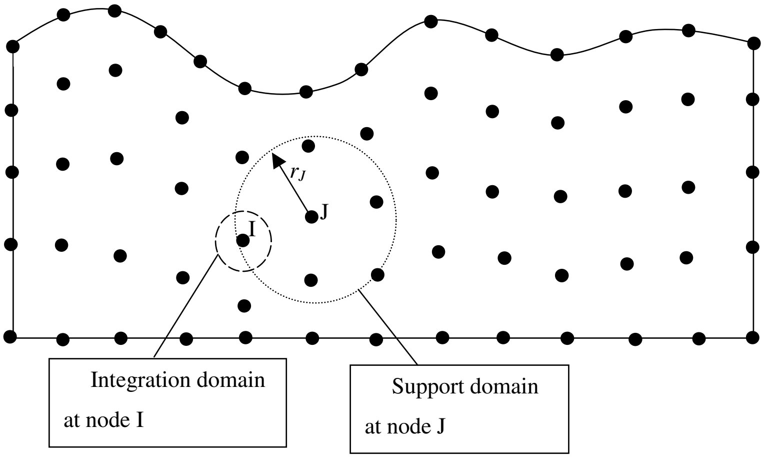


Fig. 1. Illustration of nodes, integration domain and support domain.

3. MLPG formulations

The key task in the above formulation is to find the solution for the pressure by solving Eq. (7). Various methods, such as finite element and finite different methods may be used but in this paper the MLPG method will be employed. This method is based on a set of nodes, as illustrated in Fig. 1, which discretise the fluid domain. Some of these nodes are located on boundaries and others lie inside the fluid domain. Formulation around inner nodes is first discussed and formulation around boundary nodes is described in Section 5. At each of the inner nodes, a circular sub-domain is specified. Eq. (7), after multiplying by an arbitrary test function φ, is integrated over the sub-domain, leading to

$$\int_{\Omega_I}\left[\nabla^2 p-\frac{\rho}{\Delta t}\nabla\cdot\vec{u}^{(*)}\right]\varphi\,\mathrm{d}\Omega=0, \tag{8}$$

where Ω_I is the area of the sub-domain centred at node I. There are many options for the test function [9]. One of these is based on the Heaviside step function given by

$$\varphi=\begin{cases}1 & \text{in } \Omega_I,\\ 0 & \text{otherwise.}\end{cases}$$

Using the Heaviside step function as the test function and applying the Gauss's theorem to Eq. (8) yield

$$\int_{\partial\Omega_I}\vec{n}\cdot\nabla p\,\mathrm{d}S=\frac{\rho}{\Delta t}\int_{\partial\Omega_I}\vec{n}\cdot\vec{u}^{*}\,\mathrm{d}S, \tag{9}$$

where $\partial\Omega_I$ is the boundary of Ω_I and $\vec{n}$ is the normal vector of $\partial\Omega_I$ pointing out of the sub-domain. It is noticed that Eq. (9) contains only boundary integrals and does not include the gradient for the intermediate velocity as in Eq. (8). The feature makes the evaluation of the integrals and therefore the whole procedure more efficient. This is a big advantage of the Heaviside step test function. The formulation based on this test function is also one of most accurate methods among all options available so far, according to numerical investigations made in [9].

4. Weight and shape functions

The unknown function p needs to be approximated by a set of discretised variables. Generally, the approximation may be written as

$$p(\vec{x}) \approx \sum_{j=1}^{N} \Phi_j(\vec{x})\hat{p}_j, \tag{10}$$

where N is the number of nodes that affect the pressure at point $\vec{x}$; $\hat{p}_j$ are nodal variables but not necessarily equal to the nodal values of $p(\vec{x})$; and $\Phi_j(\vec{x})$ interpolation functions called shape functions as they play a similar role to shape functions in finite element methods. However in finite element methods, the shape functions are formulated by assuming that the unknown function is described by an explicit function depending on the size of elements and connectivity between nodes and elements. In meshless methods, there are no elements and no connectivity between nodes stored. Therefore, the formulation of shape function here is entirely different from that for finite element methods. In general, a local approximation to the unknown function is assumed in meshless methods, which is expressed in terms of unknown variables corresponding to some randomly located nodes nearby. This local approximation may be formulated in a variety of ways. One of them is to use a moving least-square (MLS) method [9], which is adopted in our work. With this method, the shape function is given by

$$\Phi_J(\vec{x}) = \sum_{m=1}^{M} \psi_m(\vec{x}) \left[\mathbf{A}^{-1}(\vec{x})\mathbf{B}(\vec{x})\right]_{mJ} = \boldsymbol{\psi}^{\mathrm{T}}(\vec{x})\mathbf{A}^{-1}\left(\vec{x}\right)\mathbf{B}_J(\vec{x}) \tag{11}$$

with the base function being $\boldsymbol{\psi}^{\mathrm{T}}(\vec{x}) = [\psi_1, \psi_2, \psi_3] = [1, x, y]$ $(M = 3)$; and the matrixes $\mathbf{B}(\vec{x})$ and $\mathbf{A}(\vec{x})$ being defined as

$$\mathbf{B}(\vec{x}) = \boldsymbol{\Psi}^{\mathrm{T}}\mathbf{W}(\vec{x}) = [w_1(\vec{x} - \vec{x}_1)\boldsymbol{\psi}(\vec{x}_1), w_2(\vec{x} - \vec{x}_2)\boldsymbol{\psi}(\vec{x}_2), \ldots], \tag{12}$$

$$\mathbf{A}(\vec{x}) = \boldsymbol{\Psi}^{\mathrm{T}}\mathbf{W}(\vec{x})\boldsymbol{\Psi} = \mathbf{B}(\vec{x})\boldsymbol{\Psi}, \tag{13}$$

where $\mathbf{W}(\vec{x})$ and $\boldsymbol{\Psi}$ are, respectively, expressed by

$$\mathbf{W}(\vec{x}) = \begin{bmatrix} w_1(\vec{x} - \vec{x}_I) & 0 & \cdots & 0 \\ 0 & & & \\ \cdots & & & \\ 0 & & & w_N(\vec{x} - \vec{x}_I) \end{bmatrix} \tag{14a}$$

and

$$\boldsymbol{\Psi}^{\mathrm{T}} = [\boldsymbol{\psi}(\vec{x}_1), \boldsymbol{\psi}(\vec{x}_2), \ldots, \boldsymbol{\psi}(\vec{x}_N)] \tag{14b}$$

which shows each column of the matrix $\boldsymbol{\Psi}^{\mathrm{T}}$ is the value of the base function $\boldsymbol{\psi}$ at a particular point. The weight function $w_I(\vec{x} - \vec{x}_I)$ may be chosen to be a spline function given by

$$w_I(\vec{x} - \vec{x}_I) = \begin{cases} 1 - 6\left(\frac{d_I}{r_I}\right)^2 + 8\left(\frac{d_I}{r_I}\right)^3 - 3\left(\frac{d_I}{r_I}\right)^4 & 0 \leqslant \frac{d_I}{r_I} \leqslant 1, \\ 0 & \frac{d_I}{r_I} > 1, \end{cases} \tag{15}$$

where r_I is the size of support domain of the weight function (see Fig. 1) and $d_I = \mid \vec{x} - \vec{x}_I \mid$ the distance between the node I and the point $\vec{x}$. In order to ensure the shape function is well defined, the number of nodes (N) affecting the concerned point must be larger than M, i.e. $N \geqslant 3$ in 2D cases.

After the pressure is obtained, the gradient of the pressure may be estimated by differentiating Eq. (10)

$$\nabla p(\vec{x}) \approx \sum_{J=1}^{N} \nabla \Phi_J(\vec{x})\hat{p}_J. \tag{16}$$

The partial derivatives of the shape function with respect to x are found by differentiating Eq. (11) [20]. Thus,

$$\Phi_{J,x} = \boldsymbol{\psi}_{,x}^{\mathrm{T}}\mathbf{A}^{-1}\mathbf{B}_J + \boldsymbol{\psi}^{\mathrm{T}}\mathbf{A}_{,x}^{-1}\mathbf{B}_J + \boldsymbol{\psi}^{\mathrm{T}}\mathbf{A}^{-1}\mathbf{B}_{J,x}, \tag{17}$$

where $\mathbf{A}_{,x}^{-1}$ is the partial derivative of $\mathbf{A}^{-1}$ with respect to x and is evaluated by $\mathbf{A}_{,x}^{-1} = -\mathbf{A}^{-1}\mathbf{A}_{,x}\mathbf{A}^{-1}$ with $\mathbf{A}_{,x} = \mathbf{B}_{,x}\boldsymbol{\Psi}$; $\mathbf{B}_J$ is Jth column of Matrix $\mathbf{B}$ and its partial derivative is estimated by

$$\mathbf{B}_{J,x} = \frac{\partial w_J(\vec{x}-\vec{x}_J)}{\partial x}\boldsymbol{\psi}(\vec{x}_J). \tag{18}$$

One can also obtain the partial derivative with respect to y in a similar way or by replacing x with y in Eq. (17). It should be noted that although the base function is a linear function of coordinate variables (x, y), the shape function and its derivative are nonlinear and continuous. The gradient of the pressure given by Eq. (16) is also continuous.

5. Imposing boundary conditions

Generally, there are two kinds of boundary conditions for the pressure in water wave problems, as shown in Eqs. (2a) and (2b): the free surface condition specifying the pressure (similar to the essential boundary condition in solid dynamics) and the solid boundary condition specifying the normal derivative of the pressure. The condition on the solid boundary was generally suggested to be imposed by applying Eq. (8) to incomplete circular sub-domains of boundary nodes [8,9]. The implementation of a essential boundary condition is not very straightforward since the unknown nodal values in MLS approach to the shape function are not physical values as pointed out above and has received a considerable amount of the research as summarized in [21]. The methods suggested include the penalty method, transformation method, collocation method and so on. These approaches works well for fixed (or with little movement) boundary problems as demonstrated by the cases in the publications, e.g. [8,9,12,13,22]. However, there is no work to show whether the approaches are suitable for water wave problems, in which the large variation of unknown free surface may occur. The penalty method was tested in this work for the problems of this kind. It was found that big errors, particularly near the free surface, can be quickly built up. The reason may be due to the fact that the addition of the penalty term inevitably produces some errors since it is not an exact representation of the free surface boundary condition. These errors may be negligible if the final solution could be found in one or a few time steps. However, they may accumulate to a significant one in several thousand time steps even though the whole scheme is stable. Although the accumulated error may be reduced by shortening the length of the time step, it inevitably increases the computational costs.

Further numerical tests were performed in this work. It has been found that a better way to impose the free surface and solid boundary conditions in water wave problems is not to use the integration form in Eq. (9) for the nodes on boundaries, instead to use the following scheme:

$$\sum_{J=1}^{N} \Phi_J(\vec{x})\hat{p}_J = p_{\text{atm}} \quad \text{for nodes on the free surface} \tag{19a}$$

and

$$\sum_{J=1}^{N} \vec{n} \cdot \nabla \Phi_J(\vec{x}) \hat{p}_J = \vec{n} \cdot \left(\vec{g} - \dot{\vec{U}}\right) \quad \text{for nodes on the solid boundary.} \tag{19b}$$

Eq. (19a) is similar to a collocation method, as used in [13] for acceleration in structural dynamic problems.

6. Discretised equations

Inserting Eq. (10) into Eq. (9) and combining the result with Eq. (19) yields

$$\mathbf{K} \cdot \hat{\mathbf{P}} = \mathbf{F}, \tag{20}$$

where

$$K_{IJ} = \begin{cases} \int_{\partial\Omega_I} \vec{n} \cdot \nabla \Phi_J(\vec{x})\, \mathrm{d}s & \text{for inner nodes,} \\ \vec{n} \cdot \nabla \Phi_J(\vec{x}_I) & \text{for nodes on solid boundaries,} \\ \Phi_J(\vec{x}_I) & \text{for nodes on the free surface} \end{cases}$$

and

$$F_I = \begin{cases} \frac{\rho}{\Delta t} \int_{\partial\Omega_I} \vec{n} \cdot \vec{u}^*\, \mathrm{d}S & \text{for inner nodes,} \\ \vec{n} \cdot \left(\vec{g} - \dot{\vec{U}}\right) & \text{for nodes on solid boundaries,} \\ p_{\mathrm{atm}}(x_I) & \text{for nodes on the free surface,} \end{cases}$$

where nodes J are those having non-zero influence on node I determined by the weight function.

7. Treatment of pressure gradient

It is well known that backward or forward finite difference schemes approximating a derivative have lower order accuracy than a central scheme. Similarly, it can be understood that the pressure gradient estimated for nodes on or near boundaries using Eq. (16) may not be as accurate as for inner nodes because the related nodes to the boundary nodes distribute on one side only. In order to enhance the accuracy of the pressure gradients near the boundaries particularly near the free surface, which is important for simulating water waves, it is suggested here that the pressure gradients after estimated by Eq. (16) be treated using the following equation:

$$\nabla p(\vec{x}) \approx \sum_{J=1}^{N} \tilde{\Phi}_J(\vec{x}) (\nabla p)_J, \tag{21}$$

where the function $\tilde{\Phi}_J(\vec{x})$ is formulated by a similar method as for the shape function $\Phi_J(\vec{x})$ discussed above but using a different weight function. The weight function used for this purpose is the Gaussian weight function (e.g. [9]) defined by

$$\tilde{w}_I(\vec{x} - \vec{x}_I) = \begin{cases} \frac{\mathrm{e}^{-(\alpha_I \bar{r})^2} - \mathrm{e}^{-\alpha_I^2}}{1 - \mathrm{e}^{-\alpha_I^2}} & \bar{r} = \frac{d_I}{r_I} \leqslant 1, \\ 0 & \bar{r} = \frac{d_I}{r_I} > 1, \end{cases} \tag{22}$$

where r_I and d_I have same meanings as before and α_I is an arbitrary coefficient controlling the shape of the weight function. The larger the coefficient, the sharper the function is. Numerical tests show that the suitable value of α_I is in a range of (2–3)r_I but the best value may need to be explored in future. Although the same weight function as in Eq. (15) may be adopted herein, our numerical tests show that it does not give as good results as Eq. (22) for the treatment of the pressure gradient.

8. Numerical validation

In this section, the numerical method described above will be validated by applying it to simulating water waves in a two dimensional tank generated by a wave maker. The sketch of the problem and coordinate system are illustrated in Fig. 2. The generated wave may be mono-chromatic or bi-chromatic depending on the motion of the wave maker $S(t)$. In order to reduce the reflection, a damping zone is added at the far end opposite to the wave maker, in which the velocity is modified by adding an artificial damping term and computed by using $\vec{u}^{(n+1)} = \vec{u}^{(*)} + \vec{u}^{(**)} - v(x)\vec{u}^{(n+1)}$, in which $v(x)$ is an artificial damping coefficient defined as

$$v(x) = \frac{1}{2}v_0\left[1 - \cos\left(\frac{\pi(x - x_d)}{L_{\mathrm{dm}}}\right)\right] \quad x \geqslant x_d, \tag{23}$$

where L_{dm} is the length of the damping zone taken as $L_{\mathrm{dm}} = 3d$; x_{d} is the x-coordinate of the left end of the damping zone; and v_0 is the magnitude of the damping coefficient. A similar form of the damping coefficient was used in [23,24] for a numerical wave tank based on potential theory using a finite element method, but combined with a Sommerfeld condition. Extensive numerical tests were carried out in those papers to choose optimum values for v_0. Those values are not necessarily optimum here as the formulation is different. Nevertheless, similar numerical investigations are not carried out in this paper since the main aim here is to validate the numerical method rather than to investigate the effectiveness of the damping zone. Instead, only one suitable value is chosen based on some numerical tests, which is $v_0 = 0.1$.

The method will be validated in two ways: (1) comparing its numerical results with analytical solutions and (2) comparing its numerical results with those obtained by a finite element analysis. It should be noted that the finite element method is based on the potential theory, in which the velocity is calculated by using the gradient of the velocity potential, and so it uses very different formulation from the method described in this paper. The details of the finite element formulation can be found in [23].

When implementing the MLPG method, one must determine the sizes of integration and support domains for all nodes. Theoretically, these sizes can be chosen arbitrarily. In practice, some factors have to be considered. The size of an integration domain is determined by $R_{\Omega I} = \varepsilon h_{1I}$, where h_{1I} is the distance between Node I and its nearest node and ε is a coefficient. In order to ensure the integration domain is large enough but always located entirely inside the computational domain, the value of ε must be in the range of

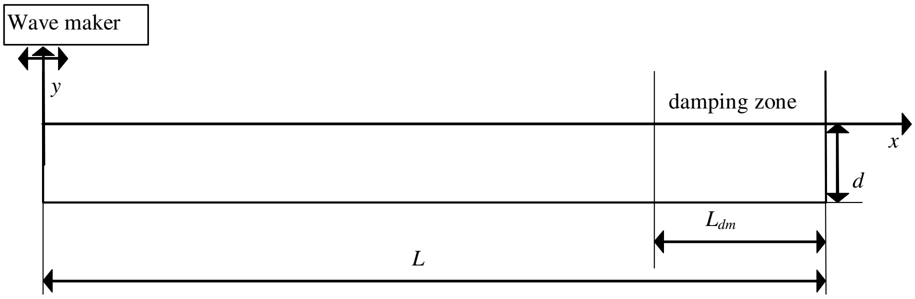


Fig. 2. Sketch of the problem and coordinate system.

$0 < \varepsilon < 1$. The size of a support domain is given by $r_I = \kappa h_{4I}$, where h_{4I} is the distance between Node I and the fourth node when counting all neighbour nodes of Node I from the nearest to the farthest one, as shown in Fig. 3. (In 2D cases with uniformly-distributed nodes, $h_{1I} = h_{4I}$.) κ is the coefficient controlling the size of a support domain and its value should be specified with care. If κ is too small, the support domain is too small and too few nodes are involved in it, which may not be enough to result in a well-defined shape function. On the other hand, if it is too large, the support domain is too big and too many nodes are contained in it. Too many nodes in one support domain require too much computational time to deal with and too large support domain may lower the accuracy. The reason for lowering the accuracy is that the linear base function used to determine the shape function can give a good approximation to the pressure field only in a small local area. When the area is too large, the approximation is inevitably degraded. Based on numerical tests, it is found that the suitable value for ε is in the range of 0.5–0.9; and the suitable value for κ is in the range of 1.5–3 and tends to be larger for larger wave amplitudes. The best values for a specific case need to be further explored. In this paper, all results are obtained using $\varepsilon = 0.6$ and $\kappa = 2.0$ for smaller amplitudes and $\kappa = 2.5$ for larger amplitudes.

In the following discussion, all variables and parameters are non-dimensionalised using d and g, i.e.

$$(x, y, L, a) \rightarrow t(x, y, L, a)d \rightarrow \tau\sqrt{\frac{d}{g}}\omega \rightarrow \omega\sqrt{\frac{g}{d}}.$$

8.1. Comparison with analytical solution

Waves with small steepness are considered first. They are generated by a piston wave maker in a tank with the length $L = 12$, undergoing the following motion

$$S(t) = S_0(1 - \cos(\omega t)) \tag{24}$$

with the amplitude $S_0 = 0.004$ and the frequency $\omega = 1.45$.

To simulate this case, nodes are uniformly distributed initially. Investigations have been carried out on effects of different numbers of nodes and different lengths of time step using $\Delta t = 0.01$ 0.02 and 0.04 and the total number of nodes $N_t = 151 \times 13 = 1963$ ($\mathrm{d}x \approx \mathrm{d}y \approx 0.08$, where $\mathrm{d}x$ and $\mathrm{d}y$ are the increment of distance between nodes in x and y direction, respectively), $172 \times 15 = 2580$ ($\mathrm{d}x \approx \mathrm{d}y \approx 0.07$) and $201 \times 17 = 3417$ ($\mathrm{d}x \approx \mathrm{d}y \approx 0.06$). It is found from these investigations that the results obtained by using $\Delta t = 0.01$ and 0.02 and $N_t = 2580$ and 3417 are very similar. Fig. 4(a) shows some typical free surface profiles obtained by different numbers of nodes with $\Delta t = 0.02$. For such a small value of the amplitude, a linearised analyt-

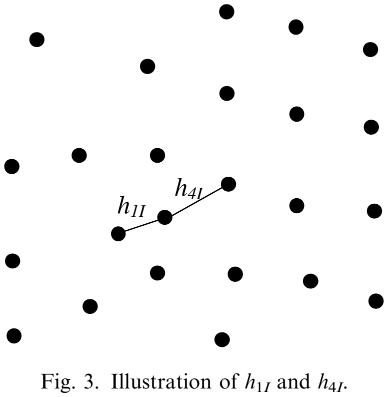


Fig. 3. Illustration of h_{1I} and h_{4I}.

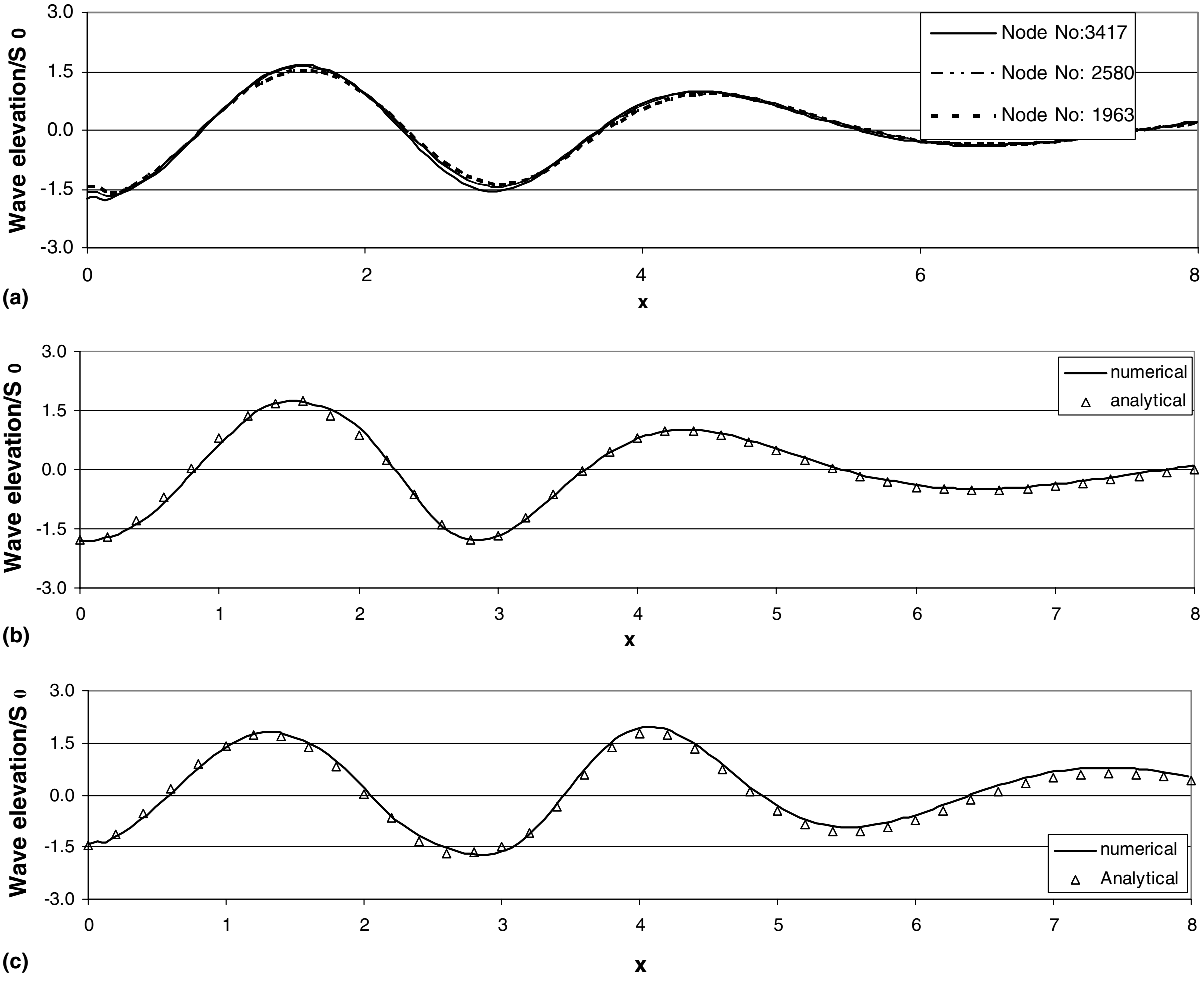


Fig. 4. (a) Wave elevation at time = 12 for $S_0 = 0.004$ and $\omega = 1.45$ obtained by different number of nodes. (b) Wave elevation at time = 12 for $S_0 = 0.004$ and $\omega = 1.45$ and comparison with analytical solution. (c) Wave elevation at time = 16 for $S_0 = 0.004$ and $\omega = 1.45$ and comparison with analytical solution.

ical solution may be found [25]. This solution is plotted in Fig. 4(b) and (c) denoted by triangles and compared with the numerical results for two different instants obtained by using $\Delta t = 0.02$ and $N_t = 3417$. It can be seen that the agreement between the numerical and analytical solutions is very good, implying that the method based on the MLPG described in this paper can produce accurate simulation of surface waves.

The performance of the MLPG method is further assessed by investigating the characteristics of relative error of the numerical results. The relative error is defined as

$$E_r = \frac{\|\eta_c - \eta_a\|}{\|\eta_a\|},$$

where $\|\eta\| = \int_{A_e} \eta^2 \, dA$ with A_e being area over which the error is estimated; η_c is the numerically computed wave elevation measured from the mean free surface and η_a is an analytical solution, such as the one given in [25]. The method used for estimating the relative error is very similar to that used in [9] but the square

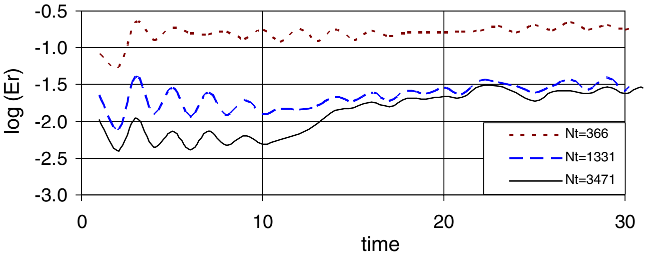


Fig. 5. The relative error at different time for different initial distance between nodes.

root of the integration is not taken because the term $\int_{A_e}\eta^2\,\mathrm{d}A$, a measurement of wave energy, possesses clearer physical meaning than $\sqrt{\int_{A_e}\eta^2\,\mathrm{d}A}$ in water wave problems. The relative error is evaluated for the case in Fig. 4 for larger increments in the node distance and presented in Fig. 5. It is observed that the relative error is oscillating with time, reflecting the basic property of water waves. This characteristic for water waves is expected and is distinguished from those given in literature (e.g. [8,9,11]) for other problems. It demonstrates that good numerical results at all time are more difficult to achieve for water waves than for other problems. Nevertheless, the error in numerical results obtained by the MLPG method is indeed reduced broadly with the decrease of distance between nodes, though the rate is different at different time. This fact implies that the numerical results with satisfactory accuracy at all time are achievable as long as a sufficiently large number of nodes are used.

8.2. Comparison with finite element methods for mono-chromatic waves

In the section, the results from the MLPG method will be compared with those from the finite element method (FEM) in [24]. The waves are still generated by the wave maker moving as specified by Eq. (24) with the same frequency.

The FEM was first employed to compute the wave elevations and to estimate the relative errors for the cases shown in Fig. 5. The mesh for the FEM is generated using the same number of nodes as in that figure, which are uniformly distributed initially. Fig. 6 depicts the comparison of the relative errors obtained from the FEM and MLPG methods for the case with the number of nodes being 3471. It can be seen from this figure that the rate of convergence of the MLPG method is faster than that of the FEM except for a short period from start. One may then deduce that the number of nodes required in the MLPG method would be smaller than that in the FEM to gain the same accuracy of numerical results under the conditions specified here. This observation happens to be similar to that described in [12], where the MLPG method was adopted to calculate the cantilever beam problems. Nevertheless, it must be noted that the characteristics

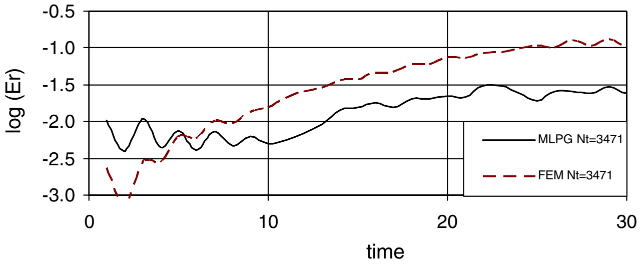


Fig. 6. Relative errors of numerical results from the FEM and MLPG methods.

of convergence of numerical results for water wave problems depend on many factors, such as the wave frequency, the wave amplitude, the distribution of nodes and so on. Further investigations are required before coming to conclude that the fact observed here generally holds in all situations.

To further check the behaviour of the MLPG-based method, it is then applied to simulating water waves with larger amplitudes. Two amplitudes are considered: one is $S_0 = 0.048$ and the other is $S_0 = 0.064$. The total number of nodes used for these cases is also 3417, which are again uniformly distributed initially. The number of nodes used for the finite element analysis is roughly similar. The comparison of results obtained by the two methods is given in Fig. 7 for $S_0 = 0.048$ and Fig. 8 for $S_0 = 0.064$, respectively. In these figures, the dots represent the positions of nodes at the time shown on the upper-right corner obtained by the MLPG-based method, with the red dots denoting the nodes constituting the free surface. The free surface resulting from the finite element method is presented by solid lines. As can be seen, the results from the two methods are very close.

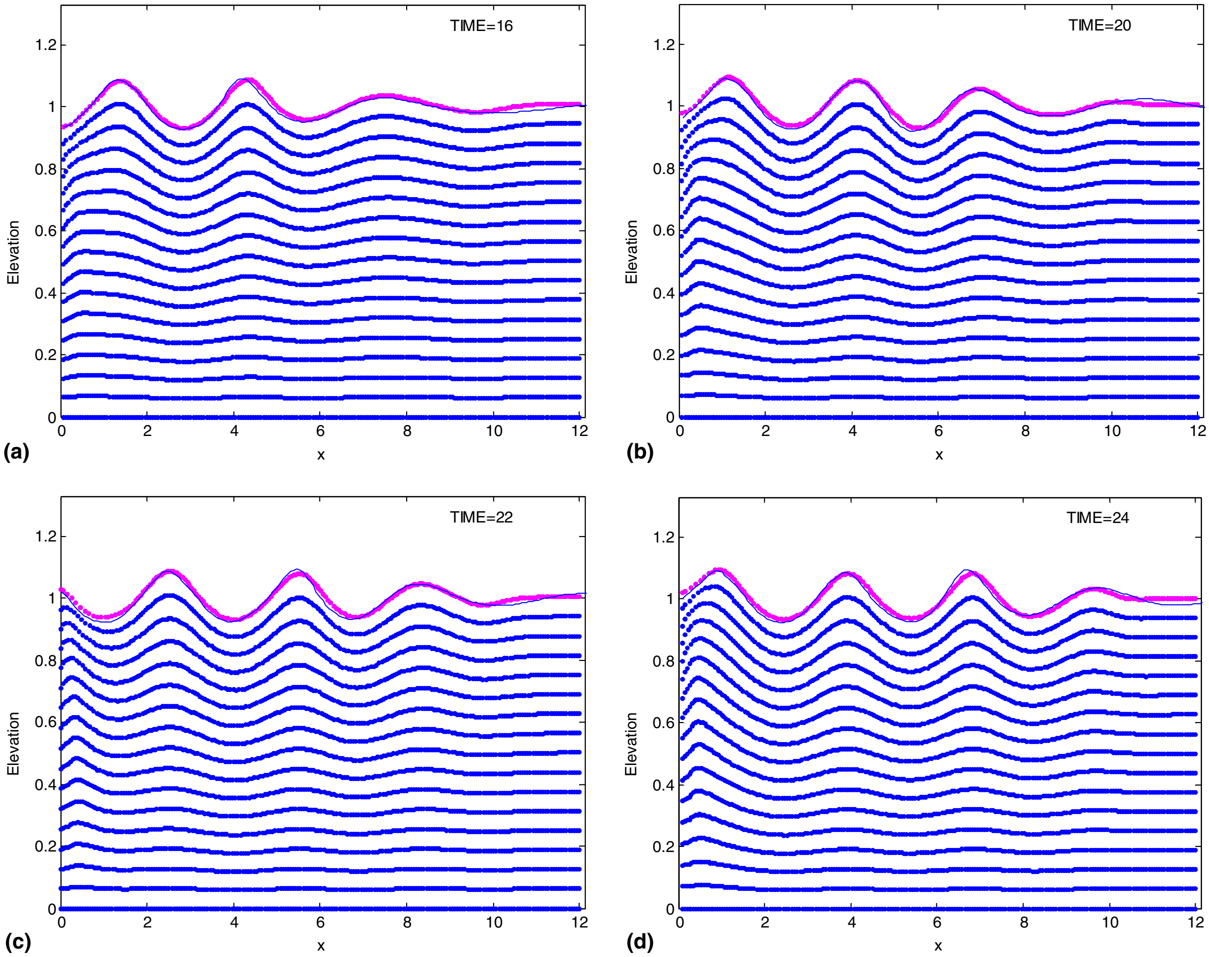
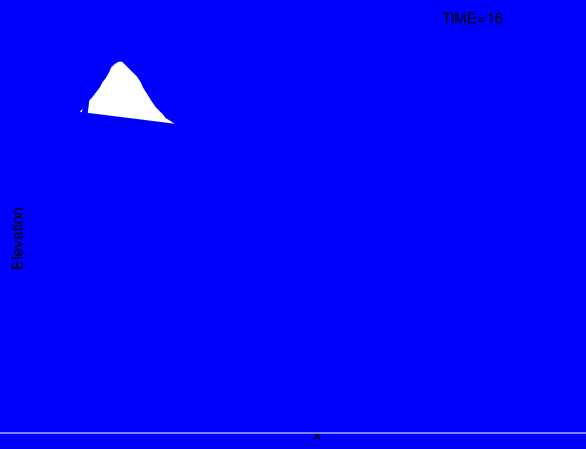
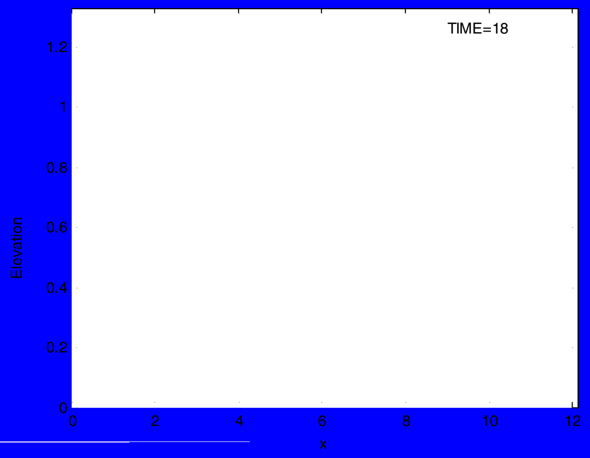
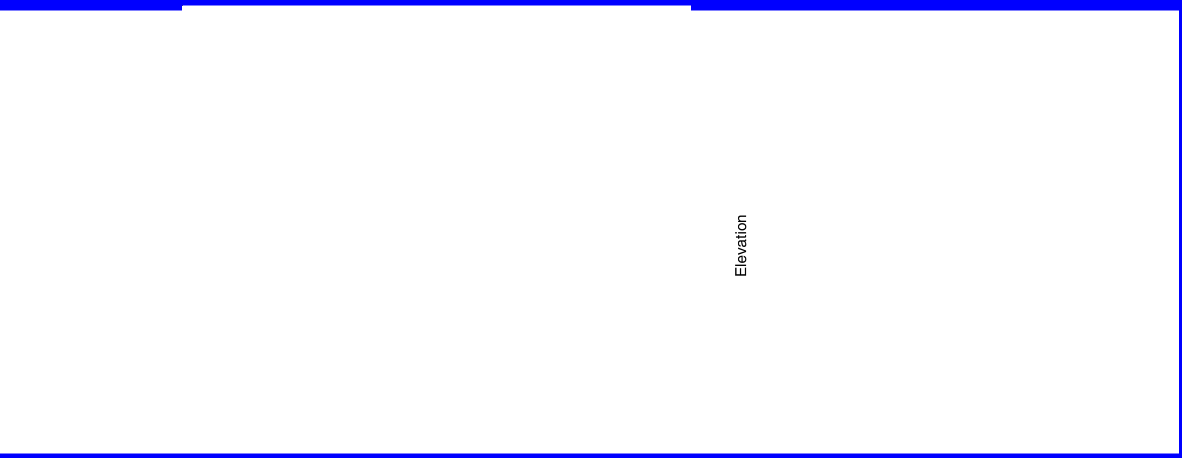
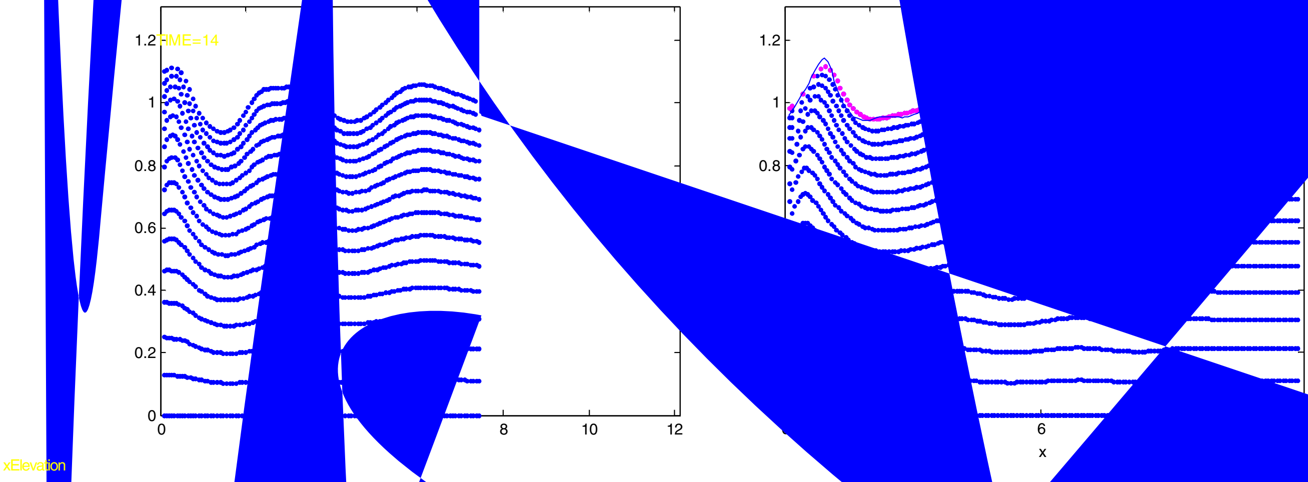


Fig. 7. Comparison of results obtained by MLPG and finite element methods for $S_0 = 0.048$ (dots, nodes; red dots, nodes constituting the free surface; solid line, the free surface obtained by the finite element method). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)


TIME=16
Elevation
x


TIME=18
1.2
1
0.8
0.6
0.4
0.2
0
Elevation
0
2
4
6
8
10
12
x


Elevation



with finite ul he agreement between ts fr
ods is also

9. Conclu

In this a meth m have been p
water wa blems. general governing
viscous f adopt . Th e value problem at each
by the m s lo rove– met newly-extended method h
two-dim a ak p resul is method agree well with ar
for linea f e sults f ly steeper waves. These comparis
that the at diffic ociated with generating and handling
as effecti ement m or water wave simulations.

Acknowl

This w sponsored , UK (GR/R78701), to which the author is most grateful

References

[1] T. Belytschko, Y. Y J. Num. M th. E

[2] B. Nayroles, C mput. Mech. 10 (19

[3] W.K. Liu of t reproduc

[4] J.J. Mo

[5] E. On overview, Int. J. Comput. Meth. 1 (2) (2004) 267

[6] R.A. Dalrymplem, O. Knio, SPH modelling of water waves, in: Proceedings of the Coastal Dynamics, Lund, 2001.

[7] E.Y.M. Lo, S. Shao, Simulation of near-shore solitary wave mechanics by an incompressible SPH method, Appl. Ocean Res. 24 (5) (2002) 275–286.

[8] S.N. Atluri, T. Zhu, A new meshless local Petrov–Galerkin (MLPG) approach in computational mechanics, Comput. Mech. 22 (1998) 117–127.

[9] S.N. Atluri, S. Shen, The meshless local Petrov–Galerkin (MLPG) method: a simple and less-costly alternative to the finite element and boundary element methods, Comput. Model. Eng. Sci. 3 (1) (2002) 11–52.

[10] R.C. Batra, H.-K. Ching, Analysis of elastodynamic deformations near a crack/notch tip by the meshless local Petrov–Galerkin (MLPG) method, Comput. Model. Eng. Sci. 3 (6) (2002) 717–730.

[11] S.N. Atluri, T. Zhu, New concepts in meshless methods, Int. J. Numer. Meth. Eng. 47 (1–3) (2000) 537–556.

[12] Z.D. Han, S.N. Atluri, Meshless local Petrov–Galerkin (MLPG) approaches for solving 3D problems in elasto-statics, Comput. Model. Eng. Sci. 6 (2) (2004) 169–188.

[13] Z.D. Han, S.N. Atluri, Meshless local Petrov–Galerkin (MLPG) approach for 3-dimensional elasto-dynamics, Comput, Mater. Continua 1 (2) (2004) 129–140.

[14] H. Lin, S.N. Atluri, Meshless local Petrov–Galerkin (MLPG) method for convection–diffusion problems, Comput. Model. Eng. Sci. 1 (2) (2000) 45–60.

[15] H. Lin, S.N. Atluri, The meshless local Petrov–Galerkin (MLPG) method for solving incompressible Navier–Stokes equations, Comput. Model. Eng. Sci. 2 (2) (2001) 117–142.

[16] A.J. Chorin, Numerical solution of Navier–Stokes equations, Math. Comput. 22 (1968) 745–762.

[17] O.C. Zienkiewicz, R.L. Taylor, The finite element method, fifth ed.Fluid Dynamics, vol. 3, Butterworth-Heinemann, 2000.

[18] S. Koshizuka, A. Nobe, Y. Oka, Numerical analysis of breaking waves using the moving particle semi-implicit method, Int. J. Numer. Meth. Fluids 26 (1998) 751–769.

[19] S.R. Idelsohn, M.A. Storti, E. Onate, Lagrangian formulations to solve free surface incompressible inviscid fluid flows, Comput. Meth. Appl. Mech. Eng. 191 (2001) 583–593.

[20] J. Dolbow, T. Belytschko, An introduction to programming the meshless element free galerkin method, Arch. Comput. Meth. Eng. 5 (3) (1998) 207–241.

[21] S.N. Atluri, The Meshless Local Petrov–Galerkin (MLPG) Method for Domain and Boundary Discretizations, Tech Science Press, 2004.

[22] E.J. Sellountos, D. Polyzos, A MLPG (LBIE) method for solving frequency domain elastic problems, Comput. Model. Eng. Sci. 4 (6) (2003) 619–636.

[23] Q.W. Ma, G.X. Wu, R. Eatock Taylor, Finite element simulation of fully nonlinear interaction between vertical cylinders and steep waves – part 1 methodology and numerical procedure, Int. J. Numer. Meth. Fluids 36 (2001) 265–285.

[24] Q.W. Ma, G.X. Wu, R. Eatock Taylor, Finite element simulation of fully nonlinear interaction between vertical cylinders and steep waves – part 2 numerical results and validation, Int. J. Numer. Meth. Fluids 36 (2001) 265–285.

[25] R. Eatock Taylor, B.T. Wang, G.X. Wu, On the transient analysis of the wavemaker, in: Ninth International Workshop on Water Waves and Floating Bodies, Kuju, Oita, Japan, 1994.