Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Computational Physics 202 (2005) 20–51
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

Proteus: a direct forcing method in the simulations of particulate flows

Zhi-Gang Feng, Efstathios E. Michaelides *

School of Engineering and Southcentral Regional Center of the National Institute for Global Environmental Change, Tulane University, New Orleans, LA 70118, USA

Received 8 January 2004; received in revised form 29 June 2004; accepted 29 June 2004
Available online 27 August 2004

Abstract

A new and efficient direct numerical method for the simulation of particulate flows is introduced. The method combines desired elements of the immersed boundary method, the direct forcing method and the lattice Boltzmann method. Adding a forcing term in the momentum equation enforces the no-slip condition on the boundary of a moving particle. By applying the direct forcing scheme, *Proteus* [1] eliminates the need for the determination of free parameters, such as the stiffness coefficient in the penalty scheme or the two relaxation parameters in the adaptive-forcing scheme. The method presents a significant improvement over the previously introduced immersed-boundary-lattice-Boltzmann method (IB-LBM) where the forcing term was computed using a penalty method and a user-defined parameter. The method allows the enforcement of the rigid body motion of a particle in a more efficient way. Compared to the "bounce-back" scheme used in the conventional LBM, the direct-forcing method provides a smoother computational boundary for particles and is capable of achieving results at higher Reynolds number flows. By using a set of Lagrangian points to track the boundary of a particle, *Proteus* eliminates any need for the determination of the boundary nodes that are prescribed by the "bounce-back" scheme at every time step. It also makes computations for particles of irregular shapes simpler and more efficient. *Proteus* has been developed in two- as well as three-dimensions. This new method has been validated by comparing its results with those from experimental measurements for a single sphere settling in an enclosure under gravity. As a demonstration of the efficiency and capabilities of the present method, the settling of a large number (1232) of spherical particles is simulated in a narrow box under two different boundary conditions. It is found that when the no-slip boundary condition is imposed at the front and rear sides of the box the particles motion is significantly hindered. Under the periodic boundary conditions, the particles move faster. The

* Corresponding author. Tel.: +1 504 865 5819/5764; fax: +1 504 862 8747.
E-mail address: emichael@tulane.edu (E.E. Michaelides).

[1] In the Greek mythology *Proteus* is a hero, the son of Poseidon. In addition to his ability to change shapes and take different forms at will, Zeus granted him the power to make correct predictions for the future. One cannot expect better attributes from a numerical code.

0021-9991/$ - see front matter
doi:10.1016/j.jcp.2004.06.020

simulations show that the sedimentation characteristics in a box with periodic boundary conditions at the two sides are very close to those found in the sedimentation of two-dimensional circular particles.

1. Introduction

Particulate flows have numerous industrial and scientific applications, such as sand storms, river sediment resuspension and transport, blood clogging and cell transport in arteries and veins, fluidized beds in chemical reactors, water treatment, etc. There are three fundamental and different approaches for the study of particulate flows: The first one is the continuum model, in which both the fluid phase and the solid phase are treated as continuous media. The continuum model requires one to know the drag coefficient and the apparent viscosity of the solid phase. The second approach is the discrete particle model. It treats the solid phase as separate particles that interact with the flow and traces the position and velocity of all the particles by solving a Lagrangian equation of motion. The fluid phase is still considered as a continuum phase, and the effects of the solid phase are included by adding mass and force terms into the continuity and momentum equations of the fluid. The third approach is the direct numerical simulation (DNS). It accounts for the solid–fluid interaction by solving the Navier–Stokes equations for the fluid phase and the initial value problem for the motion of the particles simultaneously [7,8]. With the rise of computer power, the DNS method is becoming a more enabling and popular approach to study complex particulate flow problems.

The conventional direct numerical simulation methods, such as the finite volume (FVM) and finite element methods (FEM) are not very efficient in simulations of particulate flows with a large number of particles. The main obstacle with these methods is the need to generate new, geometrically adapted grids, a very time-consuming task especially in three-dimensional flows. Methods such as Stokesian dynamics (SD) that was developed by Brady and Bossis [4] and the boundary element method (BEM) essentially neglect the fluid inertia effects and can only be applied to particulate flows under creeping flow conditions. Kalthoff et al. [18] proposed a method that incorporates analytical solutions for the region near the particle surface, with some parameters determined by matching the outer flow conditions. Hence, they are able to determine the forces on particles based on the analytical solutions. More recently, Zhang and Prosperetti [28] adopted the same idea in the ''PHYSALIS'' method. It is evident that this method requires the existence of accurate analytical solutions for the motion of particles. Such transient solutions are only available for particles with simple shapes at creeping flow conditions [22].

Ladd [19,20] successfully applied the Lattice Boltzmann method (LBM) to particle–fluid suspensions. The LBM overcame the limitations of the conventional finite volume and finite element methods by using a fixed, non-adaptive (Eulerian) grid system to represent the flow field. Since then, the LBM has proven to be a robust and efficient method to accurately simulate particulate flows with a small or a large number of particles [2,5,10,11,21].

When the LBM is used to simulate particle–fluid interaction problems, the no-slip condition on the particle–fluid interface is realized by the so-called ''bounce-back'' rule [20], and the particle surface is represented by the so-called ''boundary nodes,'' which are essentially a set of mid-points of the links between two fixed grids, in which one of the grid points is within the fluid domain and the other is within the solid domain. This arrangement causes the computational boundary of a particle to be defined by step-wise scheme. In order to represent a smooth boundary and to accurately represent the shape of any particle, it is necessary to use a large number of lattice points. In addition, when a particle moves, its computational boundary changes and varies in each time step. This introduces fluctuations in the forces that act on the

particle and limits the ability of LBM to solve particle–fluid interaction problems at very high Reynolds numbers.

Peskin [24] developed the immersed boundary method (IBM) in order to model the flow of blood in the heart. This method uses a fixed Cartesian mesh for the fluid, which is composed of Eulerian nodes. However, for the solid boundaries, which are immersed in the fluid, the IBM uses a set of Lagrangian boundary points that are advected by the fluid–solid interactions. This method is especially suitable for the simulation of the effect of deformed immersed boundaries and has been widely used in biological fluid dynamics. Fogelson and Peskin [12] have showed that this method may also be employed to simulate flows with suspended particles, whether they are deformable or not.

Höfler and Schwarzer [17] presented a finite-difference method for particle-laden flows by adding a constraint force into the Navier–Stokes equations to enforce rigid particle motions, with the constraint force being determined by a penalty method. Goldstein et al. [15] used the so-called adaptive or feedback forcing scheme to model the no-slip conditions on a stationary boundary. This technique necessitates the use of two free parameters that must be chosen, based on the flow conditions. In the recent years, the concept of IBM has been employed into the FEM. Glowinski et al. [13,14] developed the fictitious domain method (FDM) by using Lagrange multipliers to enforce the rigid body rotation and the no-slip boundary condition between the particle surfaces and the fluid. They were able to apply this method in order to simulate a flow system with 1024 spherical particles [14]. Among the other approaches, the group leaded by Tryggvason has successfully applied the control volume method with front-tracking to the solution of a large number of bubbles in multiphase flow. Many impressive results from this method appear in a recent review [27].

Ten Cate et al. [26] used an adaptive-forcing scheme with the LBM to simulate the sedimentation of a single sphere in an enclosure. Feng and Michaelides [9] combined the IBM and the LBM by computing the force density through a penalty method in the simulations of particulate flows. In this method that has been named IB-LBM, the particle boundary is treated as a deformable medium with high stiffness. Thus, a small distortion of the particle boundary yields a force that tends to restore the particle into its original shape. The balance of such forces, together with the other external forces exerted on the particle are distributed into the Eulerian nodes of the grid and the flow fields with a body force are solved over the whole fluid–particle domain by using the LBM. This method has the disadvantage that it requires the a priori selection of the stiffness parameter, based on the specific problem to be solved.

The key point of the success of both LBM and IBM is that, instead of re-meshing the fluid domain, they both use a fixed mesh to represent the fluid field. In the LBM, the moving boundaries are approximated by the fixed points on the grid, which are essentially the midpoints of the boundary links if the bounce-back rule is used to implement the no-slip boundary condition. Hence, the moving boundaries are described by Eulerian points. In the IBM, the moving boundaries are normally represented by a set of Lagrangian boundary points, which are advected by the fluid.

In this paper, we develop a new computational method called *Proteus*, which combines the direct forcing and the lattice Boltzmann methods. *Proteus* makes use of Eulerian lattice nodes for the fluid flow field and Lagrangian boundary nodes to represent particles or moving-boundary surfaces. Unlike the penalty method we have employed in our previous study [9], this method applies the direct forcing scheme, which was originally proposed by Mohd-Yusof [23] for fixed complex boundaries. This adoption eliminates the need for the determination of the free parameter for the stiffness coefficient and makes the method much more straightforward and efficient. In addition, *Proteus* allows us to implement the rigid-body conditions inside a particle in a more convenient manner.

In the first part of this paper, we give a brief description of the method, and the particle-collision rules we have used. We then implement the method to solve particulate flow problems in three-dimensions. For the validation of the method, we simulate a sphere settling in an enclosure and compare the results with the experimental measurements taken by ten Cate et al. [26]. Then we study a single particle settling under various flow conditions. Finally, we apply *Proteus* to simulate the sedimentation of 1232 spherical particles

under two different boundary conditions: the first is the no-slip boundary conditions and a small gap at the front and rear sides; the second is periodic boundary conditions at the front and rear sides.

2. The *Proteus* numerical method

2.1. Description of the method

A key feature of *Proteus* is that it resolves the no-slip boundary conditions in the numerical scheme by adding a force density term in the Navier–Stokes equations. It computes the force density via a direct forcing scheme, and then solves for the flow field by using the well-established processes of the LBM. The latter method, which was developed in the late 1980s and early 1990s, has been successfully used in several types of particulate flows by Ladd [19,20], Feng and Michaelides [10,11] and many others. The basic concept of the LBM is to decompose the flow domain into a regular lattice grid and to model the fluid as a group of fluid particles that are only allowed to move between lattice nodes or stay at rest. In order to apply the conventional LBM to particulate flows, the boundary of the solid particles is realized in the method by using the ''bounce-back rule,'' according to which, the fluid particles will bounce back when they run into a solid boundary.

One of the difficulties with this approach is that it uses boundary nodes, which are at the midpoints of boundary links. We will call these nodes Eulerian boundary nodes since they are fixed with respect to the spatial coordinates. However, since the particles move in the fluid matrix, the Eulerian nodes that represent the surface of the particles change after every time step. As a result, the actual computational boundaries of the particles change at each time step. This representation of a surface causes significant fluctuations of the computational boundaries, especially when a relatively small number of lattice nodes are used to represent the surface of the particles.

The determination of the Eulerian boundary nodes that represent the surface of the particles is a non-trivial task especially when the particles do not have simple shapes. However, the worst disadvantage of the ''bounce-back'' rule is that, at higher Reynolds numbers, it either fails to achieve accurate results or would not converge. This disadvantage requires a finer updating scheme for particle velocity and position during one lattice time step.

In order to solve this problem one may use the IB-LBM [9] and represent the particles' surfaces by using a set of independent Lagrangian boundary points that are attached to the boundary of the particles. We call these the Lagrangian boundary nodes (or points) to differentiate them from the Eulerian points described above. The Lagrangian boundary nodes represent the surface of the particles and move inside the Eulerian matrix as the particles move inside the fluid. The result is that the computational boundary of the particles is smooth and that the exact locations of the Lagrangian boundary points are easily determined if one keeps track of the transformation matrix. This constitutes a major advantage of the IB-LBM [9].

In the IBM, the existence of a solid body is represented by its effect on the fluid. This is enforced by introducing a fluid body force density term into the momentum equations. Let us consider a particle with a boundary surface, Γ, immersed in a three-dimensional incompressible viscous fluid with a domain, Ω . The particle boundary surface, is represented by the Lagrangian parametric coordinates, $\boldsymbol{s}$, and the flow domain, Ω, is represented by the Eulerian coordinates $\boldsymbol{x}$. Hence, any position on the particle surface may be written as $\boldsymbol{x} = \boldsymbol{X}(\boldsymbol{s}, t)$. Let $\boldsymbol{F}(\boldsymbol{s}, t)$ and $\boldsymbol{f}(\boldsymbol{x}, t)$ represent the particle surface force density and the fluid body force density. Then the no-slip boundary condition is satisfied by enforcing the velocity at all boundaries to be equal to the velocity of the fluid at the same location:

$$\frac{\partial \boldsymbol{X}(\boldsymbol{s}, t)}{\partial t} = \boldsymbol{u}(\boldsymbol{X}(\boldsymbol{s}, t), t), \tag{1}$$

where $\boldsymbol{u}$ is the fluid velocity.

The governing equations for the fluid–particle composite are as follows:

$$\rho\left(\frac{\partial \boldsymbol{u}}{\partial t}+\boldsymbol{u}\cdot\nabla\boldsymbol{u}\right)=\mu\nabla^2\boldsymbol{u}-\nabla p+\boldsymbol{f}, \tag{2}$$

$$\nabla\cdot\boldsymbol{u}=0, \tag{3}$$

$$\boldsymbol{f}(\boldsymbol{x},t)=\int_\Gamma \boldsymbol{F}(\boldsymbol{s},t)\delta(\boldsymbol{x}-\boldsymbol{X}(\boldsymbol{s},t))\mathrm{d}\boldsymbol{s} \tag{4}$$

and

$$\frac{\partial \boldsymbol{X}}{\partial t}=\int_\Omega \boldsymbol{u}(\boldsymbol{x},t)\delta(\boldsymbol{x}-\boldsymbol{X}(\boldsymbol{s},t))\mathrm{d}\boldsymbol{x}\ , \tag{5}$$

where p $(\boldsymbol{x}, t)$ is the fluid pressure ρ is the fluid density and μ the fluid viscosity.

Eqs. (2) and (3) are the Navier–Stokes equations for viscous incompressible flow. Eq. (4) shows how the force density of the fluid, $\boldsymbol{f}(\boldsymbol{x}, t)$, may be obtained from the immersed boundary force density, $\boldsymbol{F}(\boldsymbol{s}, t)$, through the integration over the immersed boundary. Eq. (5) is essentially the no-slip condition at the interface, since the particle moves at the same velocity as the neighboring fluid.

In the numerical implementation of the IBM the whole fluid domain, including the parts that are occupied by immersed bodies, is divided into a set of fixed regular nodes. Since these fluid nodes are not moving with the flow, we will call them Eulerian nodes. The immersed boundary of the particles is discretized by a group of boundary points that move under the action of the fluid. We will call these boundary nodes Lagrangian nodes. It must be pointed out that in the IBM, the Lagrangian nodes do not necessarily coincide with the Eulerian nodes.

In order to solve the fluid field with a body force density, $\boldsymbol{f}(\boldsymbol{x}, t)$, the fundamental LBM equation is modified by adding a term to the collision function and becomes as follows:

$$n_i(\boldsymbol{x}+\boldsymbol{e}_i,t+1)-n_i(\boldsymbol{x},t)=-\frac{1}{\tau}\left[n_i(\boldsymbol{x},t)-n_i^{(0)}(\boldsymbol{x},t)\right]+\frac{3}{2}w_i\boldsymbol{f}\cdot\boldsymbol{e}_i. \tag{6}$$

Here, $n_i(\boldsymbol{x},t)$ is fluid particle distribution function in the ith direction, $n_i^{(0)}(\boldsymbol{x},t)$ is the equilibrium distribution function, τ is the relaxation time, and t is the lattice simulation time. In this paper, a three dimensional 15 bit (3D15) LBM model has been used, and the notation for the subscripts are as follows: $i = 1$–6 corresponds to the fluid particles moving to their four nearest neighbors along the axial directions. The values $i = 7$–13 signify that the fluid particles move to their neighbors along the diagonal directions. Finally, $i = 0$ correspond to fluid particles being at rest. The vectors $\boldsymbol{e}_i$ represent the above fifteen directions. The details of the implementation of this method may be found in Feng and Michaelides [10]. The parameters w_i are weighting factors (or coefficient constant) that depend on the LBM model chosen to be used. For example the weighting factors for a 2-D problem are: $w_0 = 4/9$, $w_1 = w_3 = w_5 = w_7 = 1/9$, and $w_2 = w_4 = w_6 = w_8 = 1/36$ [19]. Finally, the dimensionless relaxation time, τ, is related to the dimensionless kinematic viscosity of the fluid by the expression:

$$v=(2\tau-1)/6. \tag{7}$$

The relaxation time that is chosen is related to the physical time step through the following relationship:

$$\Delta t=(\Delta x)^2\frac{v}{v_\mathrm{r}}, \tag{8}$$

where Δx is the physical grid step, and v_r is the physical fluid kinematic viscosity.

It is known that the numerical error in the LBM is proportional to the square of a computational Mach number, *Ma* [19]. Therefore, it is important to choose a relaxation time that keeps the Mach number small, that is:

$$Ma \equiv \frac{U_c}{\frac{1}{\sqrt{3}}(\Delta x/\Delta t)} = \frac{U_c \Delta x}{v_r}\left(\frac{2\tau - 1}{2\sqrt{3}}\right) \ll 1, \tag{9}$$

where U_c is the characteristic velocity of the actual flow.

As mentioned above, in the *Proteus* method we use a set of Lagrangian boundary points to describe the particle boundary. Eq. (2) is also valid at these Lagrangian boundary points and, hence, the force density at these points may be written as follows:

$$\boldsymbol{f} = \rho\left(\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{t}} + \boldsymbol{u}\cdot\nabla\boldsymbol{u}\right) - \mu\nabla^2\boldsymbol{u} + \nabla\boldsymbol{p}. \tag{10}$$

If we assume that the velocity and pressure fields at the time step $t = t_n$ are known, then we have an explicit scheme to determine the force term at these Lagrangian boundary points at time $t = t_{n+1}$, which is as follows:

$$f_i^{(n+1)} = \rho\left(\frac{u_i^{(n+1)} - u_i^{(n)}}{\Delta t} + u_j^{(n)}u_{j,i}^{(n)}\right) - \mu u_{i,jj}^{(n)} + p_{,i}^{(n)}. \tag{11}$$

In the last equation, the Einstein notation for subscripts and derivatives is used.

In order to impose the boundary condition that at $t = t_{n+1}$, the velocity on the immersed Lagrangian boundary points is equal to the velocity of the particle at the same point, $U_i^{P(n+1)}$, the density force at these points should be given by the following expression:

$$f_i^{(n+1)} = \rho\left(\frac{U_i^{P(n+1)} - u_i^{(n)}}{\Delta t} + u_j^{(n)}u_{j,i}^{(n)}\right) - \mu u_{i,jj}^{(n)} + p_{,i}^{(n)}. \tag{12}$$

The above equation is called direct forcing, since it may be used to evaluate the force density at the Lagrangian boundary points without introducing any pre-defined parameters. In the original presentation of the method by Mohd-Yusof [23] and Fadlun et al. (2000), instead of creating a set of Lagrangian boundary points, the force density was computed for a regular node. This node was a point where the velocity was defined when a staging grid was used, or simply a velocity point next to the boundary. The flow velocity at this point is obtained by interpolating the velocities at its neighboring velocity points and the intersection point of the solid boundary and a grid line. By doing this, one is able to achieve a second-order accurate direct-forcing scheme. However, in the case of particulate flows where the particle boundary is constantly moving, it becomes a formidable task to compute the intersection points between the particle boundary surfaces and the grid lines.

Fig. 1 shows the boundary points for a two-dimensional particle boundary. The velocities at these Lagrangian points at the current time step $t = t_n$ may be computed using Eq. (5). Alternatively, as used in the simulations by *Proteus*, one may employ a bilinear interpolation using the velocity values of four neighboring grid points for two dimensional flows or eight neighboring grid points for three dimensional flows.

The calculated force density using Eq. (12) is at a Lagrangian boundary point, and we have to spread it into the neighboring Eulerian nodes using a spreading function. In the adaptive forcing scheme, the fluid density force is also computed at boundary points. Goldstein et al. [15] explained this spreading as the smoothing the boundary surface and used a Gaussian function to smoothen the boundary within one grid

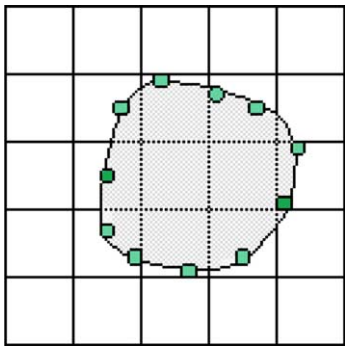

Fig. 1. A set of Lagangran boundary points for a two-dimensional particle.

spacing. In this work, we use a delta function, which will be discussed later, to spread the force density to the nearby Eulerian nodes.

The use of the force spreading technique is consistent with the theory of the immersed boundary method. The procedure employed may be explained as follows.

For simplicity, we consider a two-dimensional problem with a particle boundary Γ. We also consider a small area ε around a Lagrangian boundary point $\boldsymbol{s}_i$, which is the coordinate corresponding to a particular Lagrangian boundary node i, as shown in Fig. 2. By integrating the force density within the small area, we obtain:

$$\int_{\varepsilon} \boldsymbol{f}(\boldsymbol{X}(\boldsymbol{s},t))\mathrm{d}A = \int_{\Gamma} \boldsymbol{F}(\boldsymbol{s},t)\left(\int_{\varepsilon} \delta(\boldsymbol{x}-\boldsymbol{X}(\boldsymbol{s},t))\mathrm{d}A\right)\mathrm{d}\boldsymbol{s} = \int_{\Gamma_\varepsilon} \boldsymbol{F}(\boldsymbol{s},t)\mathrm{d}s. \tag{13}$$

The relation between the flow force density, $\boldsymbol{f}(\boldsymbol{X}(\boldsymbol{s}, t))$, and the surface force density, $\boldsymbol{F}(\boldsymbol{s}, t)$, of Eq. (4) was used in Eq. (13). The latter implies that the flow force density integral for a small area ε is equal to the boundary force density integral over the boundary element Γ_ε, which is the intersection of this small area and the whole particle boundary Γ (from point Γ_1 to point Γ_2 in Fig. 2). We envision that the flow force on the body is distributed in a small area along the particle boundary, Γ, with a force density given by the function $\boldsymbol{f}(\boldsymbol{X}(\boldsymbol{s}, t))$. In the numerical implementation, we consider a small area of this band represented by a Lagrangian boundary node $\boldsymbol{s}_i$. Then the flow force density on this area is approximated by the value at this boundary node, $\boldsymbol{f}(\boldsymbol{X}(\boldsymbol{s}_i, t))$. For a uniform grid with grid spacing $\delta x = \delta y = \delta$, it is reasonable to assume that this "area of influence" represented by the node $\boldsymbol{s}_i$ is $\delta s\delta$ (where δs is the length of the small

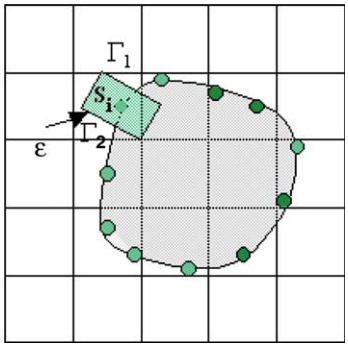

Fig. 2. A schematic for the force spreading.

boundary element Γ_ε). Then the force acting on this area is $\boldsymbol{f}(\boldsymbol{X}(\boldsymbol{s}_i t))\delta s\delta$ and the integral of the surface force density over the small area may be approximated as

$$\int_{\Gamma_\varepsilon} \boldsymbol{F}(\boldsymbol{s},t)\mathrm{d}s \approx \boldsymbol{f}(\boldsymbol{X}(\boldsymbol{s}_i,t))\delta_s\delta. \tag{14}$$

For the three dimensional case, one needs to consider a small volume around a Lagrangian point. Hence, the line integral should be replaced by a surface integral over a small boundary surface area intersected by the small volume. Also, δs should be replaced by δA, the area of this boundary surface element.

By using Eq. (14) as an approximation for the surface force density integral, the flow force density at each Eulerian node is determined from Eq. (4). This is the same procedure for the force spreading as if a delta function were used for the spreading of the force. Eq. (14) also provides an approach for the computation of the force acting on the particles, since the total force acting on a particle is equal to the sum of the forces acting on each surface element.

A separate method to compute the force acting on the particles is by integrating the fluid stress over the particle surface:

$$F_i^{P(n)} = \oint_\Gamma \sigma_{ij}^{(n)} n_j \,\mathrm{d}S = \oint_\Gamma \left(-p^{(n)}\delta_{ij} + \mu\left(u_{i,j}^{(n)} + u_{j,i}^{(n)}\right)\right) n_j \,\mathrm{d}S. \tag{15}$$

When the direct forcing method is also implemented for the internal fluid, so that the internal fluid follows a rigid-body motion, then the surface integral is taken only over the external surface. However, when the internal fluid is allowed to be freely developed, the surface integral should take into consideration both the external and the internal surface. After calculating the forces on particles by using the summation of the forces on each element and the direct integration of the shear force over the particle surface, we have found that there is a negligible difference between the computed values by using these two different approaches.

It must be pointed out that these spreading techniques are only accurate to the first-order, which is the same order of accuracy of the bounce-back rule in the conventional LBM implementation. Nevertheless, the method still provides reasonably accurate results for particulate flows, as will be seen in the examples that will follow.

2.2. Three-dimensional implementation

The direct-forcing method combined with LBM is implemented in three-dimensional particulate flows. The first issue arising is how to set up the Lagrangian boundary points in order to accurately represent the particle surface. For a spherical particle in three dimensions, it is impossible to find evenly distributed boundary nodes that represent the surface of the sphere. For this reason, we use a number of strips, with the width of strip being comparable to the grid spacing. Each strip is composed by a number of evenly distributed points. The number of the points in each strip is chosen in a way that the spacing between two neighboring points is approximately equal to the width of the strips. In the simulations we conducted in this paper, the surface area of a Lagrangian boundary point is typically not larger than two in lattice units.

As we mentioned earlier, the forces are computed at the Lagrangian boundary nodes. For the fluid to be affected by them, the forces need be spread to the Eulerian nodes. For this reason, the force corresponding to each Lagrangian boundary node is spread into its neighboring Eulerian nodes. If the spreading Eq. (17) is used, this spreading occurs within a distance of two lattice units from each Lagrangian node. This process is performed for each Lagrangian boundary node, and the force on a Eulerian node is the summation of all the contributions from these Lagrangian boundary nodes. The spreading process can be stated mathematically as follows: For a system of N number of particles, with M Lagrangian boundary points used to represent each particle surface, the flow force density at a Eulerian node $\boldsymbol{x}_i$ may be computed as follows:

$$\boldsymbol{f}(\boldsymbol{x}_i,t)=\sum_{n=1}^{N}\sum_{m=1}^{M}\boldsymbol{f}(\boldsymbol{x}_{mn}(t),t)\delta A_{mn}D(\boldsymbol{x}_i-\boldsymbol{x}_{mn}(t)), \tag{16}$$

where $\boldsymbol{x}_{mn}$ is the position of the mth Lagrangian boundary point for the nth particle at the time t, δA_{mn} is the surface represented by boundary point, $\boldsymbol{x}_{mn}$, and the function $D(\boldsymbol{r})$ is a continuous kernel distribution function that approximates the delta function. The choice of this function must meet certain criteria, which were outlined by Peskin [25]. For the three-dimensional simulations the following equations are used:

$$\delta(r)=\begin{cases}\frac{1}{4}\left(1+\cos\left(\frac{\pi|r|}{2}\right)\right), & |r|\leqslant 2,\\ 0, & |r|>2\end{cases} \tag{17}$$

and

$$D(\boldsymbol{x}-\boldsymbol{x}_{mn})=\delta(x-x_{mn})\delta(y-y_{mn})\delta(z-z_{mn}). \tag{18}$$

When *Proteus* is used to simulate the particle motion in the fluid it is necessary to compute the force and torque on each particle. The force on a particle includes forces such as the gravity/buoyancy force as well as the particle collision forces, $\boldsymbol{F}_i^{\mathrm{col}}$. Hence, the total force exerted on the ith particle may be written as follows:

$$\boldsymbol{F}_i=\left(1-\frac{\boldsymbol{\rho}_{\mathrm{f}}}{\boldsymbol{\rho}_{\mathrm{s}}}\right)M_i\boldsymbol{g}+\oint_{S_i}\boldsymbol{\sigma}\cdot\boldsymbol{n}\,\mathrm{d}S+\boldsymbol{F}_i^{\mathrm{col}}, \tag{19}$$

where M_i is the mass of the ith particle. It must be pointed out that the LBM as well as *Proteus* compute the total drag force and not its two components, the form and the viscous drag. It is well documented in the literature that the LBM computes accurately the total drag force. We conducted several simulations on the total force using the conventional LBM as well as *Proteus* and found out that the two methods are consistent and give identical results for the drag force.

The torque on a particle is computed by using the following expression:

$$\boldsymbol{T}_i=\oint_{S_i}(\boldsymbol{x}-\boldsymbol{x}_i)\times(\sigma\cdot\boldsymbol{n})\mathrm{d}S. \tag{20}$$

In order to minimize the skewing effect (drift) due to the accumulation of the numerical error, when one uses the rotation matrix of the particle, a unit quaternion is used to represent the rotation of the particles. In our three dimensional implementation, we use a Runge–Kutta scheme to solve the following set of differential equations:

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix}\boldsymbol{X}(t)\\ \boldsymbol{q}(t)\\ \boldsymbol{P}(t)\\ \boldsymbol{L}(t)\end{bmatrix}_i=\begin{bmatrix}\boldsymbol{v}(t)\\ \frac{1}{2}\varpi(t)\boldsymbol{q}(t)\\ \boldsymbol{F}(t)\\ \boldsymbol{T}(t)\end{bmatrix}_i, \tag{21}$$

with initial conditions given as

$$\begin{bmatrix}\boldsymbol{x}(0)\\ \boldsymbol{q}(0)\\ \boldsymbol{P}(0)\\ \boldsymbol{L}(0)\end{bmatrix}_i=\begin{bmatrix}\boldsymbol{x}_{i,0}\\ \boldsymbol{q}_{i,0}\\ m_i\boldsymbol{v}_{i,0}\\ \boldsymbol{I}_i\varpi_{i,0}\end{bmatrix}, \tag{22}$$

where, $\boldsymbol{q}_i$(t) is the unit quaternion form for the rotation matrix of ith particle, with its initial value is determined from the initial rotation matrix; $\boldsymbol{P}_i$(t) is the linear momentum of the ith particle; and $\boldsymbol{L}_i(t)$ is the angular momentum of the ith particle.

At any given time, the translational and rotational velocities of a spherical particle are computed by using the following expressions:

$$\boldsymbol{v}_i(t) = \frac{\boldsymbol{P}_i(t)}{M_i}; \qquad \boldsymbol{\varpi}_i(t) = \boldsymbol{I}_i(t)^{-1}\boldsymbol{L}_i(t), \tag{23}$$

where $\boldsymbol{I}_i$(t) is the inertia matrix of the ith particle in the current coordinate system. This is related to the moment of inertia matrix I_{i0} of the initial coordinate system of the ith particle and the rotation matrix of the ith particle, R_i(t), by the expression:

$$\boldsymbol{I}_i(t) = \boldsymbol{R}_i(t)\boldsymbol{I}_{i0}\boldsymbol{R}_i(t)^{\mathrm{T}}. \tag{24}$$

The numerical scheme employed is capable to simulate the motion and rotation of particles with any arbitrary shape. In the special case of spherical particles, $\boldsymbol{I}_i$(t) is a constant diagonal matrix:

$$\boldsymbol{I}_i = \begin{bmatrix} \frac{2}{5}M_i r_i^2 & & \\ & \frac{2}{5}M_i r_i^2 & \\ & & \frac{2}{5}M_i r_i^2 \end{bmatrix}, \tag{25}$$

in which r_i the radius of ith particle.

One of the disadvantages of the conventional LBM is that when the bounce-back rule used in particulate flows, the rigid body motion inside the particle is not enforced a priori. Therefore, the problem that is solved is actually the approximate interaction between a fluid and a solid shell. The shell has the same boundary as the particle, but the contribution of the interior mass of the fluid is ignored. Generally it is believed that this contribution from the internal flow is not significant, as discussed by Ladd (2001). In the present method with the direct-forcing scheme, we treat the particles as volume-less shells filled with the same fluid. Since the spreading of force at the boundary nodes is not limited to the flow outside the particles, the internal flow is allowed to be developed. To remedy this deficiency, we may enforce the rigid body motion for the internal fluid by setting up a certain number of internal Lagrangian points. The velocity at these points is determined by the rigid body motion of the particle. Hence, one of the advantages of *Proteus* is that it enforces the rigid body motion for the interior flow in particles in a straightforward way. This is a desired attribute of any method that handles solid particles. This fact notwithstanding, it must be pointed out that in our simulations of a single particle sedimentation where the flow is non-rotational we have confirmed the observation of Ladd (2001) that the effect of the internal flow on the particle motion is not significant. It would be of interest to employ *Proteus* and further examine this effect with different types of flows and higher Reynolds numbers.

2.3. Particle collision technique

In any type of particulate flows collisions between particles are unavoidable, especially when the flow is dense and the particles move at high Reynolds numbers. The correct handling of these collisions in any direct numerical simulation (DNS) is very important for the study of all particulate processes. Generally, the grid used in a DNS study is not fine enough to handle the lubrication force that develops between the particles or between particles and a solid boundary. Therefore, an artificial mechanism is necessary to be introduced in the numerical scheme in order to account for the repulsive force during collision processes. Without such a mechanism, it is likely that the particles will penetrate significantly into each other's computational boundary, thus, rendering the results meaningless.

In our previous study [9], we employed a collision technique by introducing a repulsive force when the gap between two particles is lower than a given threshold, the so-called "safe zone." This artificial short-range repulsive force is added as an external force, with the functional form that was developed by Glowinski et al. [14]:

$$\boldsymbol{F}_{ij}^{\mathrm{P}} = \begin{cases} 0, & \left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\| > R_i + R_j + \zeta, \\ \frac{c_{ij}}{\varepsilon_{\mathrm{P}}}\left(\frac{\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\| - R_i - R_j - \zeta}{\zeta}\right)^2 \left(\frac{\boldsymbol{x}_i - \boldsymbol{x}_j}{\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|}\right), & \left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\| \leqslant R_i + R_j + \zeta. \end{cases} \tag{26}$$

The parameter, c_{ij} is the force scale. This is chosen to be the buoyancy force on the particle for the sedimentation problems considered here. Of the other parameters, ε_{P} is the stiffness parameter for collisions, R_i and R_j are the radii of two particles, respectively, and ζ is the threshold or "safe zone," which is specified in advance. Similarly, the repulsive force between a particle and a wall is given by the reflection method as follows:

$$\boldsymbol{F}_{ij}^{\mathrm{W}} = \begin{cases} 0, & \left\|\boldsymbol{x}_i - \boldsymbol{x}_{i,j}\right\| > 2R_i + \zeta, \\ \frac{c_{ij}}{\varepsilon_{\mathrm{W}}}\left(\frac{\left\|\boldsymbol{x}_i - \boldsymbol{x}_{i,j}\right\| - 2R_i - \zeta}{\zeta}\right)^2 \left(\frac{\boldsymbol{x}_i - \boldsymbol{x}_j}{\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|}\right), & \left\|\boldsymbol{x}_i - \boldsymbol{x}_{i,j}\right\| \leqslant 2R_i + \zeta, \end{cases} \tag{27}$$

where $\boldsymbol{x}_{i,j}$ is the position of a fictitious particle $P_{i,j}$ which is located in a symmetric positions at the other side of the wall W_j and ε_{W} is another stiffness parameter. Glowinski et al. [14] provided the justification and an extensive discussion on how to choose these stiffness parameters.

This collision technique allows particles to overlap when the stiffness parameter ε_{P} is very large, that is, when the particles undergo "soft" collisions. For "soft" collisions, the partial overlapping of particles may be significant when a large number of particles undergo a packing process. The particles at the bottom, which have to bear the load of the particles above, will exhibit the maximum overlapping. To counteract this overlapping, one has to choose a higher value for the repulsive force when the collision scheme given in Eq. (26) is used. This results in "hard collisions," which are accompanied by undesirable side effects.

To resolve this issue, we employ a new collision scheme in the present paper. This new approach chooses the repulsive force by considering the following situations: before the two particles contact, a repulsive force given by Eq. (26) is used; when the two particles start to overlap, a higher spring force is applied. This force is proportional to the overlapping distance of two particles, and typically is much larger than the repulsive force without overlapping. For two spherical particles, this collision force has the following form:

$$\boldsymbol{F}_{ij}^{\mathrm{P}} = \begin{cases} 0, & \left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\| > R_i + R_j + \boldsymbol{\zeta}, \\ \frac{c_{ij}}{\varepsilon_{\mathrm{P}}}\left(\frac{\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\| - R_i - R_j - \zeta}{\zeta}\right)^2 \left(\frac{\boldsymbol{x}_i - \boldsymbol{x}_j}{\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|}\right), & R_i + R_j < \left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\| \leqslant R_i + R_j + \boldsymbol{\zeta}, \\ \left(\frac{c_{ij}}{\varepsilon_{\mathrm{P}}}\left(\frac{\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\| - R_i - R_j - \zeta}{\zeta}\right)^2 + \frac{c_{ij}}{E_{\mathrm{P}}}\frac{\left(R_i + R_j - \left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|\right)}{\zeta}\right)\left(\frac{\boldsymbol{x}_i - \boldsymbol{x}_j}{\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|}\right), & \left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\| \leqslant R_i + R_j, \end{cases} \tag{28}$$

in which E_{P} is a smaller parameter than ε_{P} to ensure a much larger spring force. With this type of force the overlapping of two particles is low. The first term in the last function of Eq. (28) is added so that the collision force will be continuous at particles gap $(R_i + R_j - \|x_i - x_j\|) = 0$.

The advantage of the present collision scheme is that it enables us to use a small repulsive force for particles sedimentation before packing and the larger spring force for particles in the packing process. When particles start the sedimentation process and before packing, the small repulsive force will be enough to repel the two particles and this will reduce unwanted side effects by the large repulsive force. When the

particles start packing, the spring force will take effect to prevent large particles penetrating among each other.

3. Validation of the method

3.1. Simulation of a sphere settling in a wide enclosure

Regarding the validation of the method, it must also be pointed out that the convergence of the computational results with respect to the LBM nodes has been well documented by many previous researchers, such as Ladd [20]. We are adding here the simulation results of the present method with respect to the number of the surface nodes. This clearly shows that for a sphere of diameter 15 lattice units, a mere 100 boundary nodes used to represent its surface will provide very good results. Thus, we compared our simulation results for a group of spherical particles settling in an enclosure, with the experimental measurements by ten Cate et al. [26] who measured the trajectories and velocities of particles experimentally using a PIV system. They also obtained numerical simulation results for the process using the LBM that was combined with an adaptive forcing scheme.

We consider the case of a spherical particle settling in a box of dimensions $10 \times 10 \times 16$ cm^3. The particle commences its motion at a height $H = 12$ cm from the bottom, as depicted in Fig. 3. The fluid density in the simulations is in the range from 960 to 970 kg/m^3, and the dynamic viscosity from 0.058 to 0.353 N s/m^2. The diameter of the particle is 15 mm and its density 1120 kg/m^3. Table 1 lists the fluid properties used in the experiments, and the Reynolds numbers based on the measured velocities, together with the parameters used in our simulations. The enclosure is simulated using $100 \times 100 \times 160$ grid points, and the particle is outlined by 15 lattice grid points; the grid step is $\Delta x = 1$ mm.

Regarding the computational cost of the current method, it is comparable to the cost of the conventional LBM. Most of this computational time is associated with the lattice nodes [10,20]. In our simulations for a single particle, we have found out that the number of surface nodes does not play a significant role in the computational time. For example, for the simulation of one particle with $d = 15$ in an enclosure of $100 \times 100 \times 160$ lattice units, the computational time increased by only 4% when the number of surface nodes increased from 68 to 262. Fig. 4 shows this comparison of the results of the method by using different

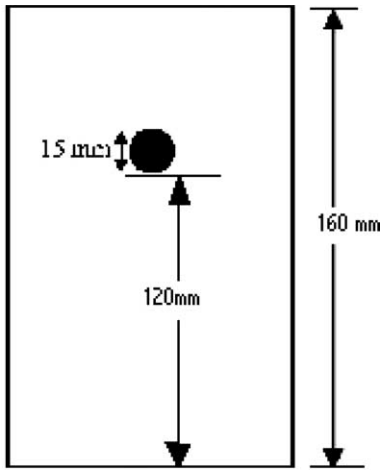

Fig. 3. Schematic diagram for a single particle settling in an enclosure.

Table 1
Fluid properties in the experiment and parameters used in simulations

	ρ_f (kg/m^3)	μ_f (10^{-3} N s/m^2)	Re	τ	Δt (10^{-4} s)
Case E1	970	373	1.5	0.9	3.47
Case E2	965	212	4.1	0.9	6.07
Case E3	962	113	11.6	0.8	8.51
Case E4	960	58	32.2	0.65	8.28

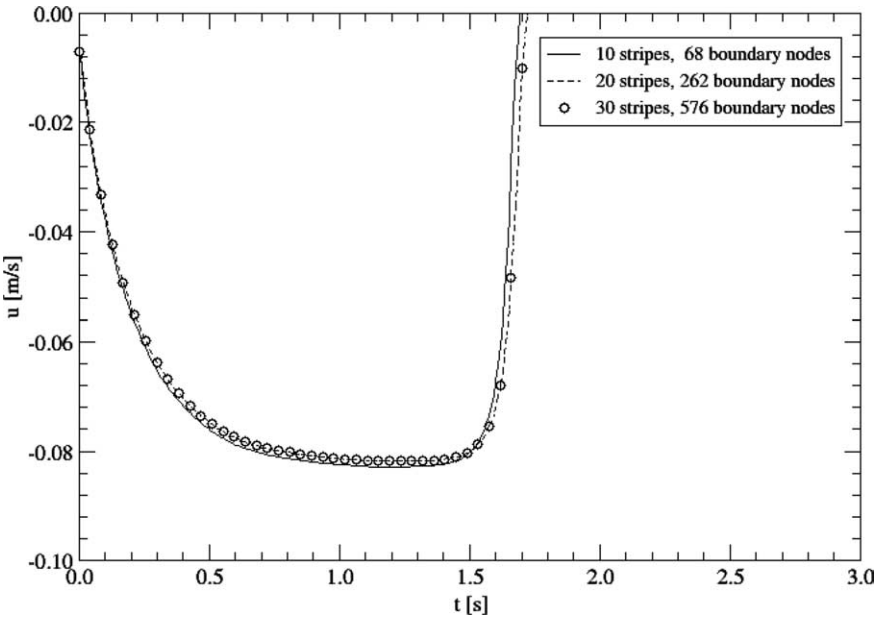

Fig. 4. Comparison of particle setting velocities at $Re = 11.6$ using a different number of Lagrangian boundary nodes.

numbers of Lagrangian points for the representation of a the surface of the particles. The figure depicts the settling velocities of a sphere for the case E3 ($Re = 11.6$), when three different numbers of surface boundary nodes 68, 262 and 572 are used. The results show not only good convergence with respect to the resolution of boundary surface nodes, but also indicate that satisfactory results may be achieved even with 68 boundary nodes. For the system examined that only has one particle, the difference in the computational cost is not significant between these three cases.

Figs. 5 and 6 show the particle trajectories and settling velocities between the experimental measurements and the numerical results from *Proteus* for four different Reynolds numbers, based on the terminal velocity of the sedimentation process, that range from 1.5 to 32.2. It is observed that the simulation results for both the particle velocity and trajectory agree well with the experimental measurements. The slowing of the particles and their final rest at the end of the process are due to the presence of the bottom wall. The simulated trajectories agree very well with the experimental measurements and the particle velocity difference is always less than 8%.

It must be pointed out that, in the numerical simulation conducted by ten Cate et al. [26] the so-called "calibrated diameter" was used for the computation of the force on the particles. This calibration is accomplished by comparing the numerical results for the drag force with analytical results. Thus, the drag force

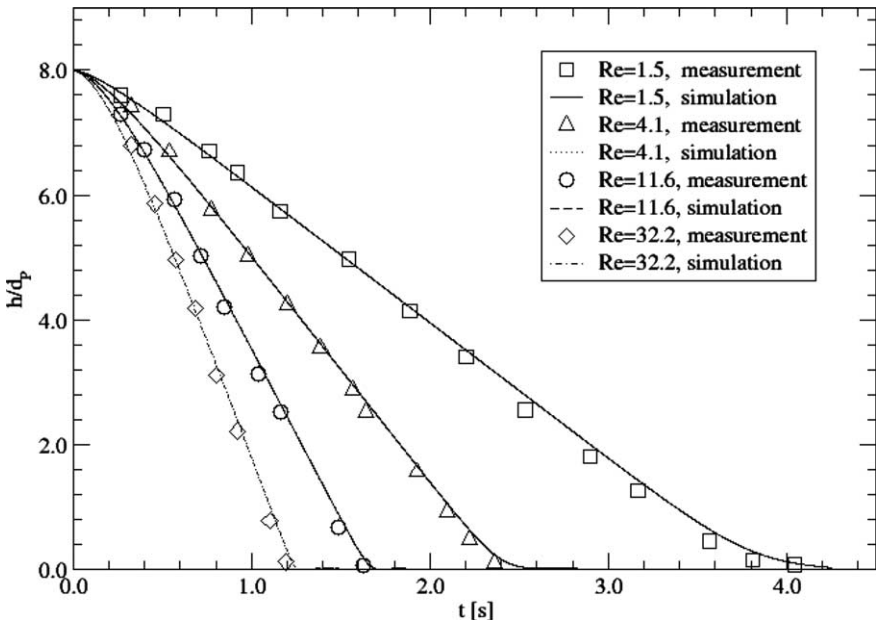

Fig. 5. Comparisons between trajectories by simulations and measurements.

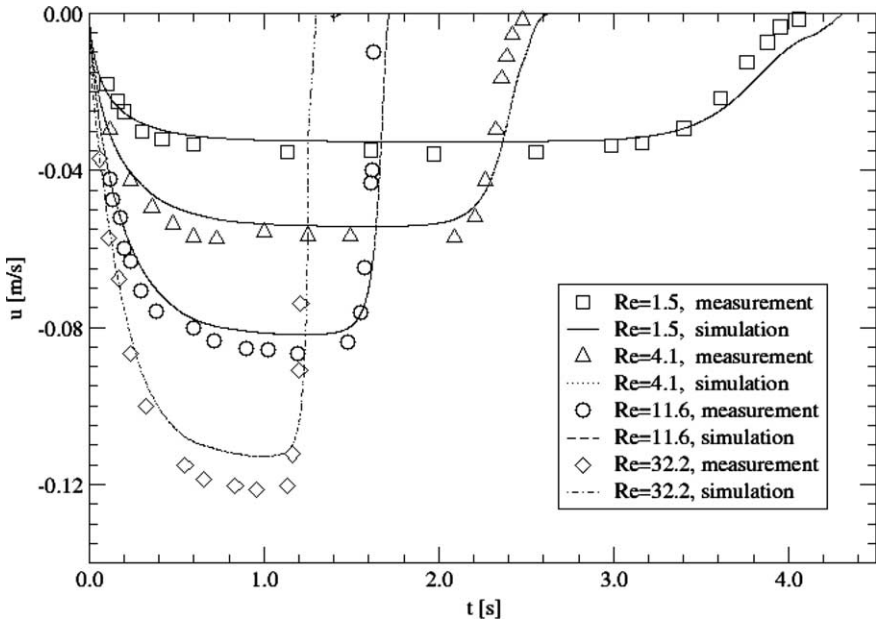

Fig. 6. Comparisons of the measured and simulated settling velocity.

on a stationary sphere in a periodic cell is obtained. Then the corresponding diameter for the sphere is computed from the analytical solution that renders the drag force equal to the one obtained from the numerical solution. An obvious drawback of this method is that by adjusting the calibrated diameter, the method also

modifies all the other computational errors regardless of their origin. Another drawback is that we only have accurate analytical solutions for the drag force at very low Reynolds number flows [22]. Therefore, the calibration is done for creeping flows, while the method may be used at high Reynolds numbers. In addition, the calibrated diameter does not only depend on the particle "input" diameter, but it also depends on other parameters of the simulation, a fact that makes the calibration process complicated and somehow vague. Because of this, it was observed by ten Cate et al. [26] that the calibrated diameters could be as much as 50% larger than the input diameters of the simulations. The simulations that are shown in the following sections do not require any calibration on the size of the particles.

3.2. Sedimentation of a sphere in a narrow enclosure

We have also modeled the sedimentation of a single particle in a narrow rectangular enclosure. This was done for the validation of the computational method as well as for estimating the accuracy of the computations when the diameter of the particles is only 8 lattice units. The dimensions of the enclosure are 1 cm length, 1 cm height, which corresponds to a grid of 128×128 lattice units. The diameter of the particle is 0.0625 cm and the width of the enclosure is variable in this computation, ranging from $1.5d$ to $3d$. The physical properties of the fluid are very close to the properties of pure water. The fluid density is $\rho_f = 1000$ kg/m^3, and the particle/fluid density ratio is 1.01. The relaxation time in this case is $\tau = 0.9915$ and each lattice time step corresponds to a physical time of 0.001 s. Hence, the kinematic viscosity of the fluid is 0.001 kg/ms.

Two different boundary conditions for the sides of the enclosure are studied:

(a) The front and rear are solid walls.
(b) The front and rear surfaces are periodic boundaries.

For each boundary condition, we perform computations for three different widths of the enclosure, $1.5d$, $2.5d$, and $3d$, respectively. Hence, a total of six cases are studied. The initial condition of the particle is at rest with its center at 0.9 cm from the bottom and at the center of the horizontal cross section.

The terminal velocity for a sphere settling in an infinite medium with the same fluid properties may be easily obtained by using an empirical equation for the drag coefficient, such as the one derived by Abraham [1]:

$$C_{d\infty} = \frac{24}{9.06^2}\left(\frac{9.06}{\sqrt{Re}} + 1\right)^2. \tag{29}$$

For the problem at hand the terminal velocity of the sphere is 0.17 cm/s, and the Reynolds number, based on the terminal velocity is $Re = 1.07$.

The effect of the walls is to increase the hydrodynamic drag and to slow down the sphere. For this problem we have several analytical and experimental data, which may serve for the validation of the computational method. The slowing may be thought of as an increase of the drag coefficient by a factor K_{wall} [16]. Since the Reynolds number in our simulation is low enough, one may use the Faxen [6] expression as an approximation for this coefficient:

$$K_{wall} = \frac{1}{1 - 1.004\lambda + 0.418\lambda^3 + 0.21\lambda^4 - 0.169\lambda^5 + \cdots}, \tag{30}$$

where λ is the ratio between the particle diameter and the width of the two parallel plates. Faxen's correction factor is applicable for $\lambda < 0.5$. For small Reynolds number flow, the drag coefficient that includes both the inertia and the wall effects is approximated by the following expression [16]:

$$C_d = C_{d\infty} + \frac{24}{Re}(K_{wall} - 1). \tag{31}$$

The coefficient C_d in the above equation is the drag coefficient for a sphere setting between two infinite plates at low Reynolds numbers. By applying this expression for the drag coefficient we calculated that the terminal velocities are 0.12 and 0.13 cm/s for gaps of $2.5d$ and $3d$, respectively. The corresponding values for the highest velocity in our simulations are 0.113 and 0.124 cm/s. Given that in the simulations we also have the effect of the bottom wall, that eventually brings the sphere to a stop, there is very good agreement between the results of the simulation and the settling process of the particle. When the width of the enclosure is $1.5d$ ($\lambda = 0.667$), the Faxen [6] formula cannot be applied and a higher-order approximation is needed. As an approximation, we used the expression by Bohlin [3], which is applicable to cylindrical enclosures. For the problem at hand and $\lambda = 0.667$, the terminal velocity in a cylindrical enclosure would be 0.062 cm/s. As would be expected, this is a bit lower than the value of 0.07 cm/s, which we obtained for the rectangular enclosure in our simulations. These comparisons show that *Proteus* applied with particles where the diameter is only 8 lattice units yields reasonable results, without the necessity for any calibration of the diameter of particles a fact that further validates the numerical method.

Fig. 7 shows the settling velocities of the 6 cases considered here as a function of time. In all the cases with solid boundaries (cases 1, 2 and 3), the particles reach steady state at approximately $t = 2.5$ s and then begin to slow down because of the presence of the bottom wall. It is seen that with the same widths of $1.5d$, the settling velocity under the no-slip boundary conditions at the sidewalls (case 1) is much smaller than the settling velocity under the periodic boundary condition (case 4). Actually, the maximum setting velocity in case 4 is slightly larger (0.19 cm/s) than the terminal velocity of the sphere in an infinite medium, which is 0.17 cm/s. This is due to the periodic boundary condition: the flow field of the implied infinite array of neighboring particles has a pulling effect on the particle and causes it to travel faster.

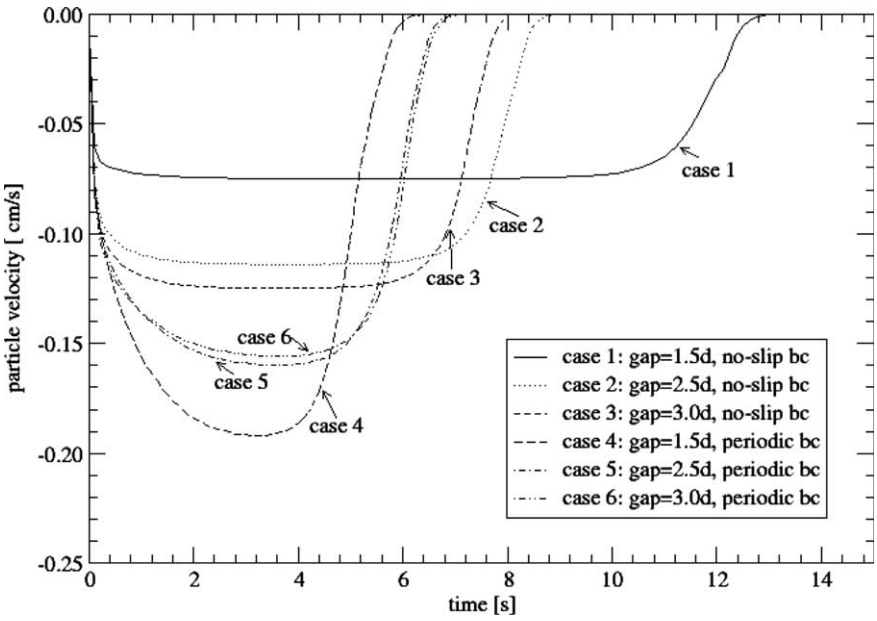

Fig. 7. The particle settling velocity as a function of time.

The velocity profiles in cases 5 and 6, which have been obtained with the periodic conditions at the sidewalls, are very close to each other indicating that the influence from the implied neighboring particles becomes insignificant when the distance between their centers is longer than $2.5d$. However, as mentioned above, with the no-slip boundary condition at the side walls, the maximum velocity depends significantly on the width of the enclosure. It is also found that the particles in cases 5 and 6 almost reach their steady state before the effect of the bottom wall becomes significant. In this case, the maximum velocity is 0.16 cm/s, a value that is very close to the terminal velocity for the particle (0.17 cm/s) in an infinite domain. All these results are additional indications that the computational method and the parameters used in this simulation yield accurate results.

Fig. 8 depicts the trajectories of the particles for the six cases considered here. Fig. 9 shows the velocity vectors at the middle cross-section of the enclosure in cases 1 and 4. The same scales for velocity vectors are used in both cases. It is apparent that the velocity field caused by the settling of the sphere is much more profound in the case where periodic boundary conditions are used.

4. Sedimentation of 1232 spherical particles in a shallow box

4.1. Setup of the problem

In this section, we extend the simulation of the last case and use *Proteus* to study the sedimentation process of 1232 spherical particles in a narrow enclosure. The initial setup of this problem is shown in Fig. 10. The group of particles is initially packed in a closed three-dimensional box, 3.125 cm long, 3.125 cm high and 0.09375 cm wide. The diameter of the particles is $d = 0.0625$ cm and, hence, the width of the box is $1.5d$. As in the previous case, the fluid density is $\rho_f = 1000$ kg/m^3, and the particle/fluid density ratio is 1.01. The kinematic viscosity of the fluid is 0.001 kg/ms. The "safe zone" between particles for these simulations was

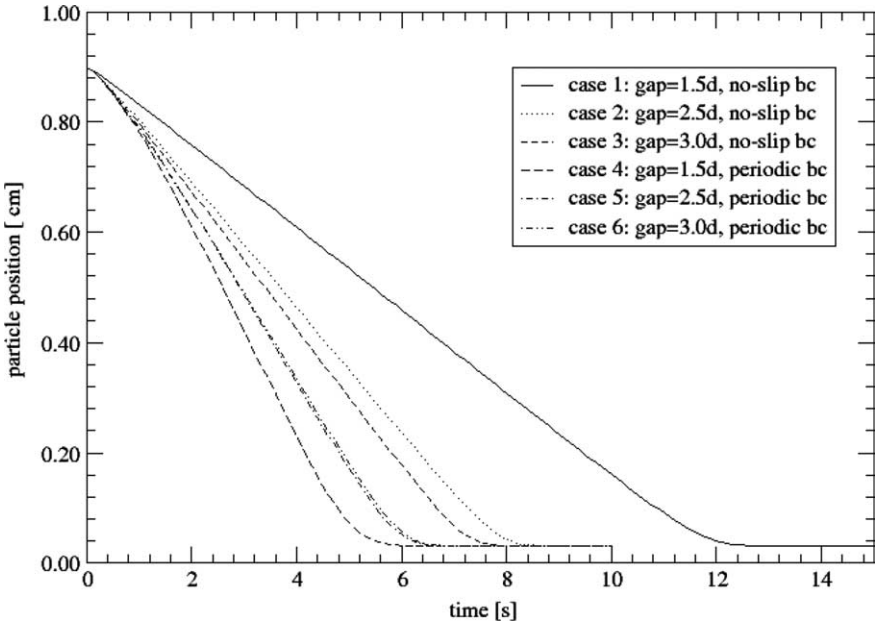

Fig. 8. The particle trajectories at various conditions.

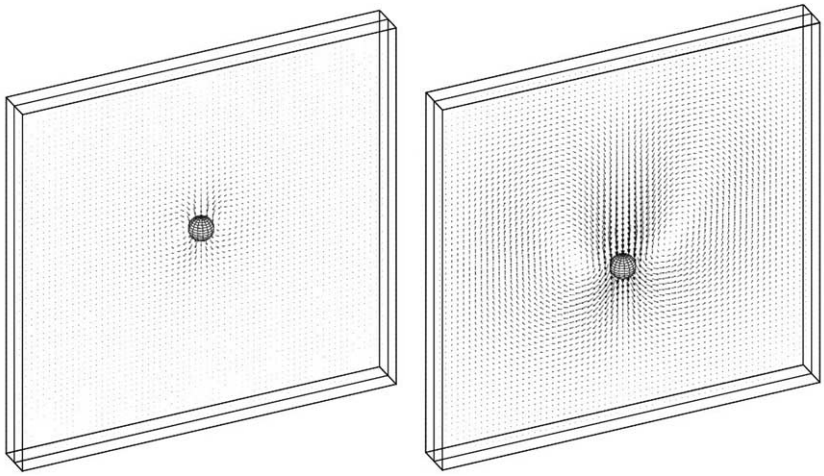

Fig. 9. Sedimentation of a sphere in an enclosure with a gap =1.5d. Left: no-slip condition at $t = 5$ s. Right: periodic boundary conditions, at $t = 3$ s. The same velocity scale is used.

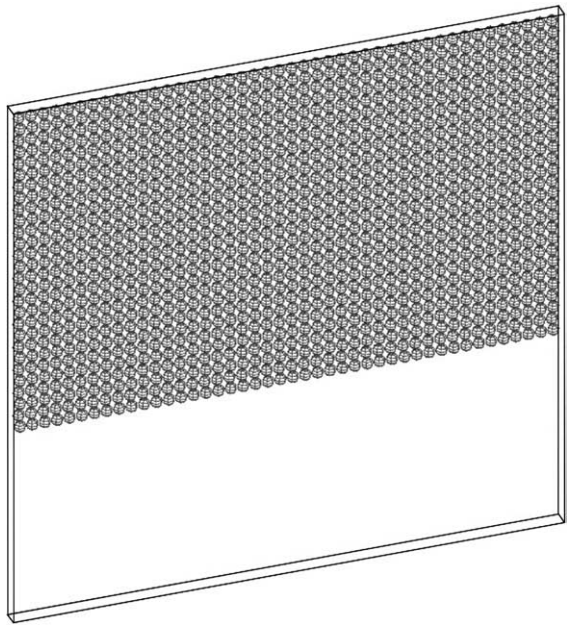

Fig. 10. Initial positions for the 1232 spherical particles. The width of the plates is 1.5d.

chosen to be equal to $d/8$. The stiffness parameters for the collisions are $\varepsilon_P = 0.25$, $E_P = 0.02$ and $\varepsilon_W = 0.5\varepsilon_P$. From the results presented in the last section, it is reasonable to conclude that *Proteus* will yield accurate results with the parameters chosen for this simulation.

Initially, both the fluid and particles are stationary in an arrangement similar to closely-packed spheres, with the heavier particles on top of the fluid. There are 28 lines of particles with each horizontal line having

44 particles. The initial gap between two neighboring particles is $d/8$. The gap between the upper wall and the first line of particles (line 1) is $3d/8$. The gap between the left wall and the left-most particle of the odd horizontal lines (lines 1, 3, 5, ...) is $3d/8$. The gap between the left wall to the left-most particle of the even horizontal lines (lines 2, 4, 5, ...) is $2d/8$. The gaps at the right side are $2d/8$ and $3d/8$, respectively, for the odd and even lines.

The numerical simulation box is $400 \times 12 \times 400$ in lattice units, and the diameter of each particle is equal to 8 lattice units. As in the last simulation, two boundary conditions will be examined in the width direction: (a) solid walls with no-slip boundary condition and (b) periodic boundary conditions, which imply an infinite array of identical particles. The boundaries in all the other directions are solid boundaries with no-slip velocity conditions. The relaxation time for the first case is $\tau = 0.9915$ and each lattice time step corresponds to a physical time of 0.001 s; the relaxation time for the second case $\tau = 0.74576$ and each lattice time step corresponds to a physical time of 0.0005 s.

All the simulations were conducted on a SGI Onyx 3500 machine. In the case of sedimentation with 1232 particles, the time to complete a single iteration is about 15.8 s. This results in approximately 4.3 h computational time to simulate 1 s of physical time for the first case, or about 11 days to complete a simulation of 60 s of physical time without any paralleling.

4.2. Sedimentation of 1232 particles with no-slip boundary conditions at all sides

As shown in the previous section, the small width between front and rear walls slows significantly the particles. Figs. 11–21 are "snapshots" of the sedimentation process for these particles and also show the flow velocity vectors at the central vertical cross-section of the flow field.

It is observed in this case with the no-slip boundary conditions applied at all sides that, the motion of the particles is significantly retarded because of the presence of the sidewalls. The whole arrangement moves downwards and the upper part of the enclosure becomes clear within 50 s from the commencement of

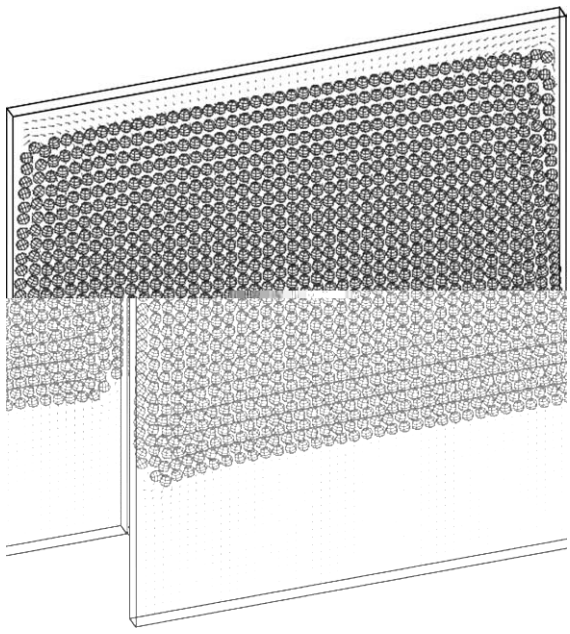

Fig. 11. Particle positions at $t = 5$ s.

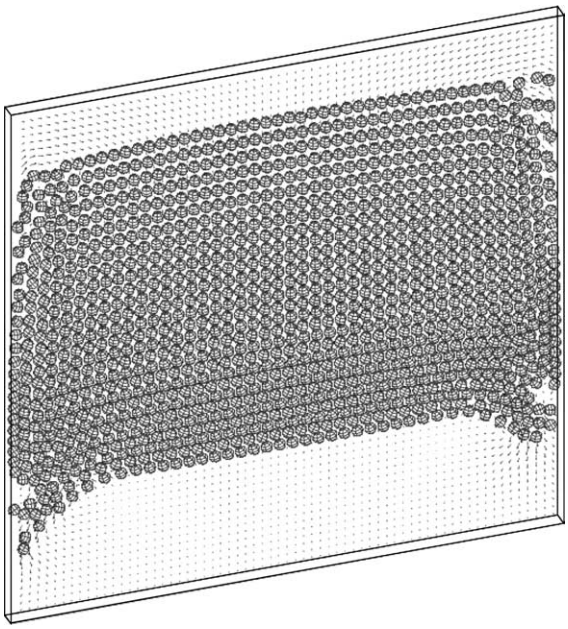

Fig. 12. Particle positions at $t = 10$ s.

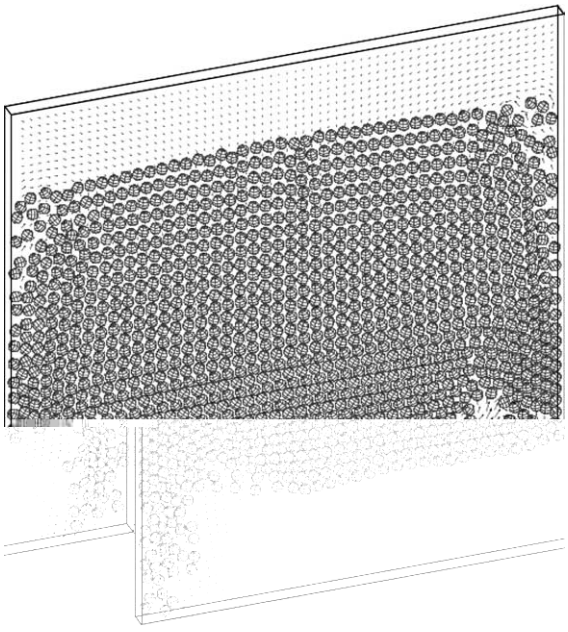

Fig. 13. Particle positions at $t = 15$ s.

the process. The groups of particles in the sides settle a little faster and reach the bottom faster on the sides of the container. Although there are such perceivable indications of a Rayleigh–Taylor type instability [9,14], its effects are not significant as in the case of the two-dimensional simulations.

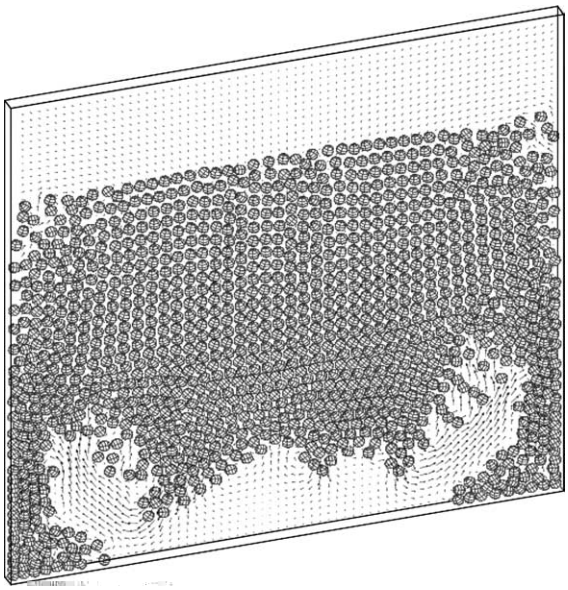

Fig. 14. Particle positions at $t = 20$ s.

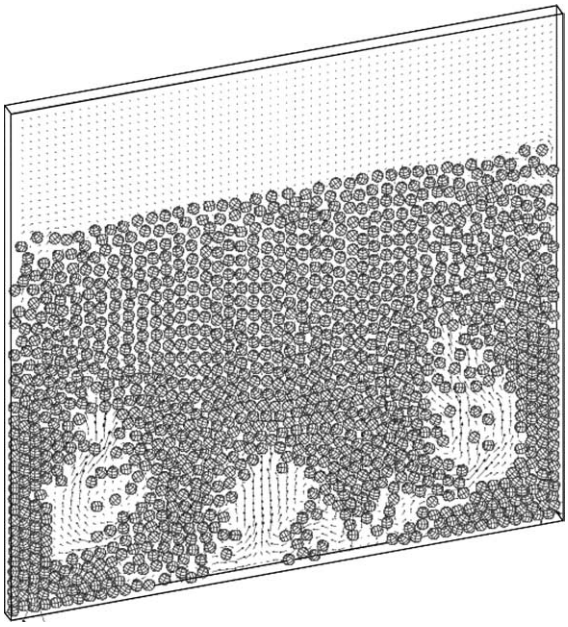

Fig. 15. Particle positions at $t = 25$ s.

4.3. Sedimentation of 1232 particles with periodic boundaries in the transverse direction

The case of the simulation of infinite arrays of three-dimensional spheres is closer to the two-dimensional case that was studied by Glowinski et al. [14] and Feng and Michaelides [9]. Periodic

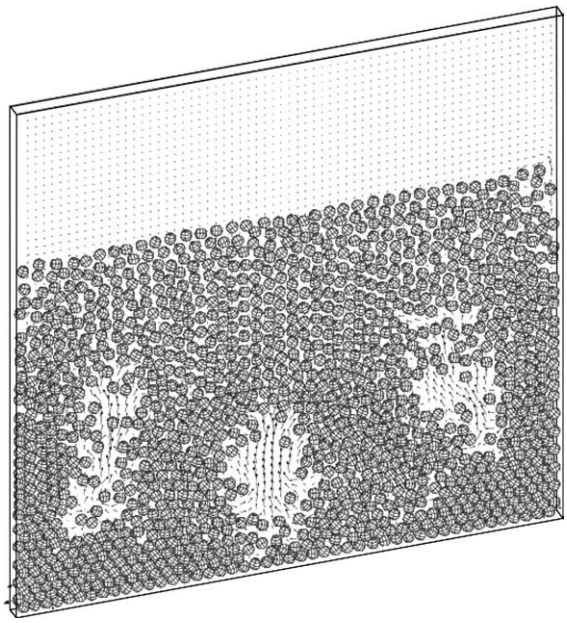

Fig. 16. Particle positions at $t = 30$ s.

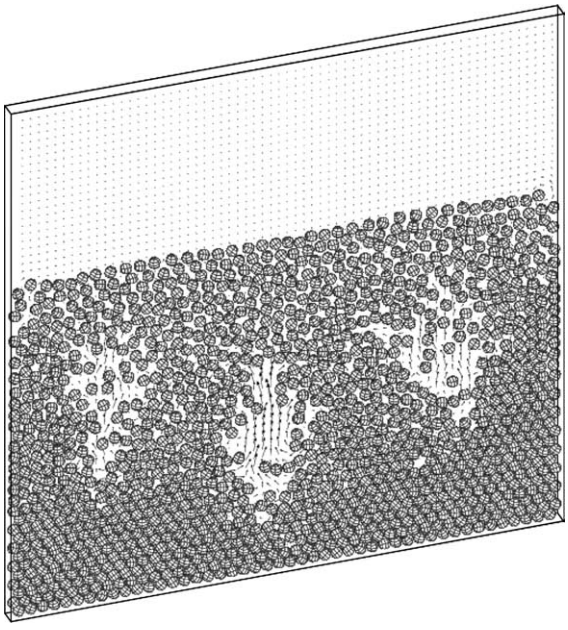

Fig. 17. Particle positions at $t = 35$ s.

boundary conditions are applied to the walls in the transverse direction that defines the narrow enclosure. The effect of these conditions is to introduce implied neighboring spheres and, in the case of a single sphere, these neighbors exert a pulling effect that causes the sphere to settle faster. Figs. 22–32 show

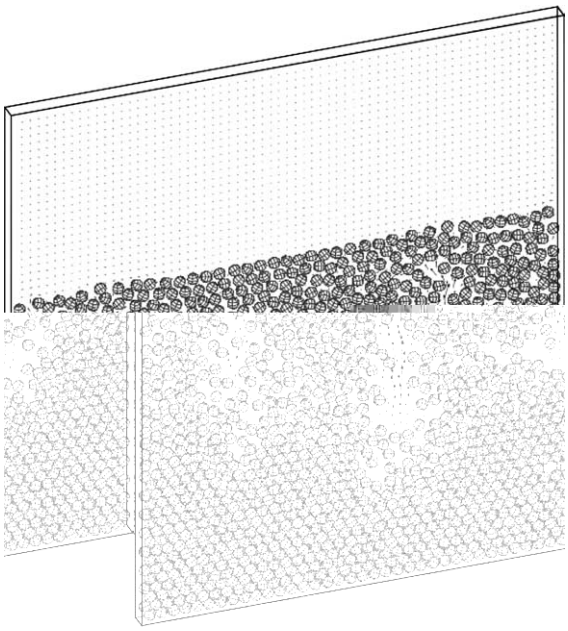

Fig. 18. Particle positions at $t = 40$ s.

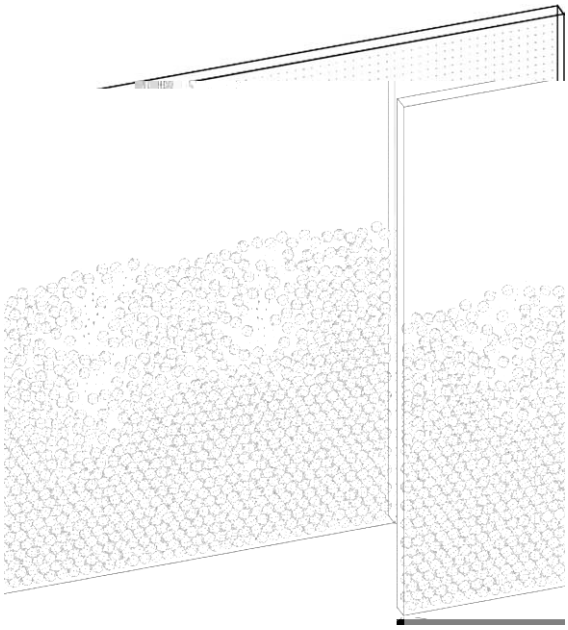

Fig. 19. Particle positions at $t = 45$ s.

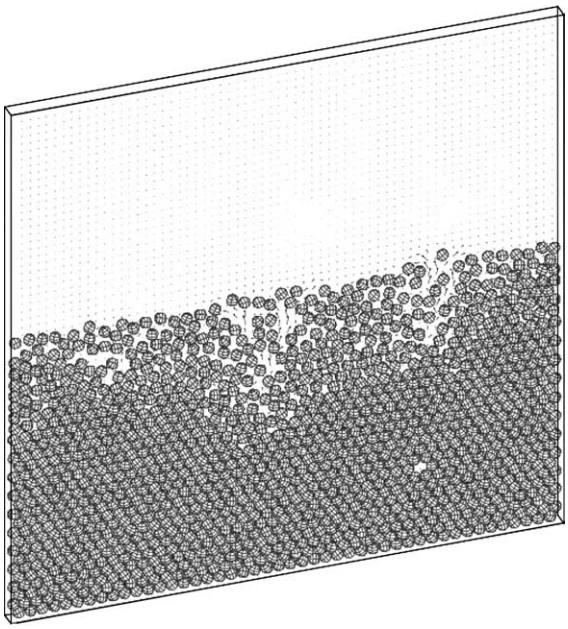

Fig. 20. Particle positions at $t = 50$ s.

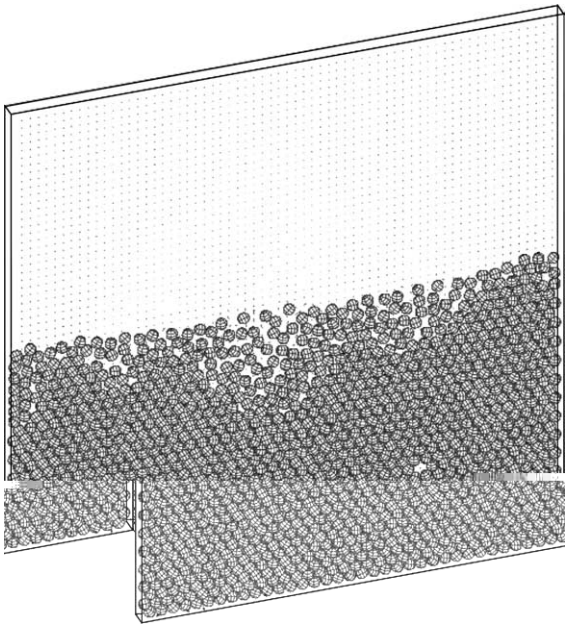

Fig. 21. Particle positions at $t = 55$ s.

a series snapshots at different times of this process, together with the velocity vector field in the central vertical cross-section. The scale used for all the velocity vectors in the figures is half of the scale used in Figs. 11–21.

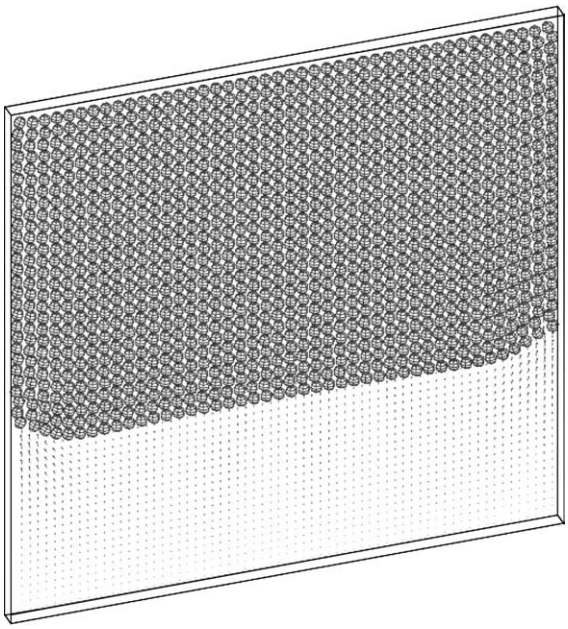

Fig. 22. Particle positions at $t = 2$ s.

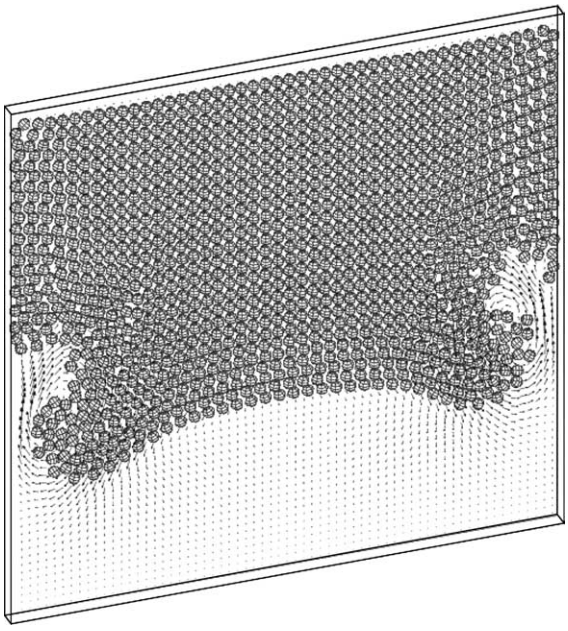

Fig. 23. Particle positions at $t = 4$ s.

It is evident in this simulation that there are very strong effects of a Raleigh–Taylor instability. Two large vortices form at the sides of the enclosure and, while they assist the particles on the sides to settle faster, they also levitate particles at the center and this motion results in the "lifting" of the particles at the middle

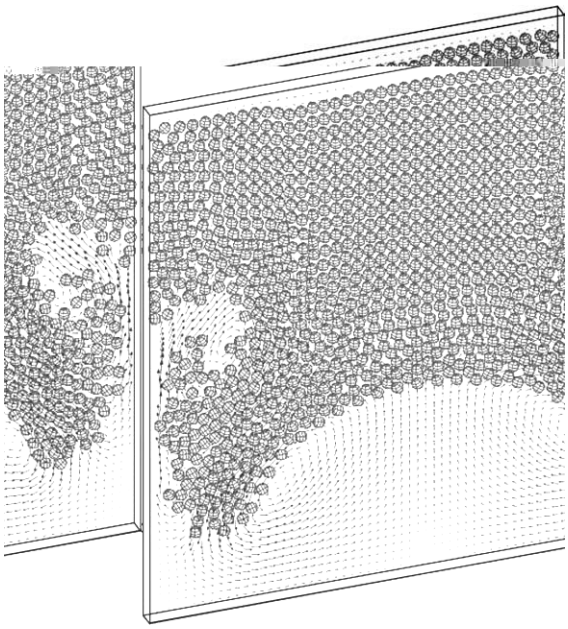

Fig. 24. Particle positions at $t = 5$ s.

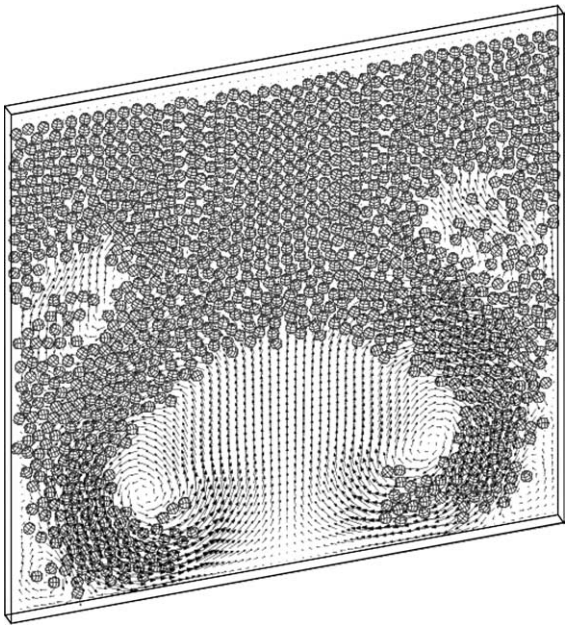

Fig. 25. Particle positions at $t = 6$ s.

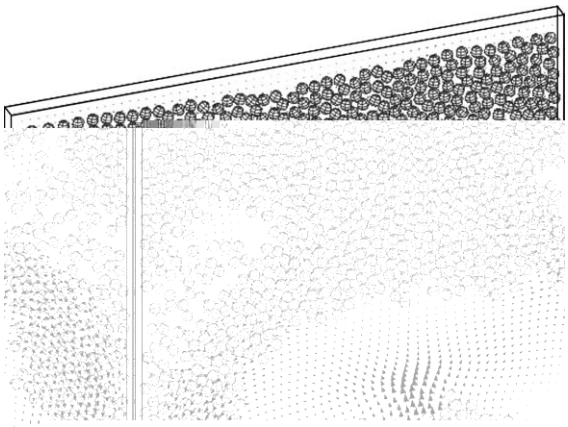

Fig. 26. Particle positions at $t = 7$ s.

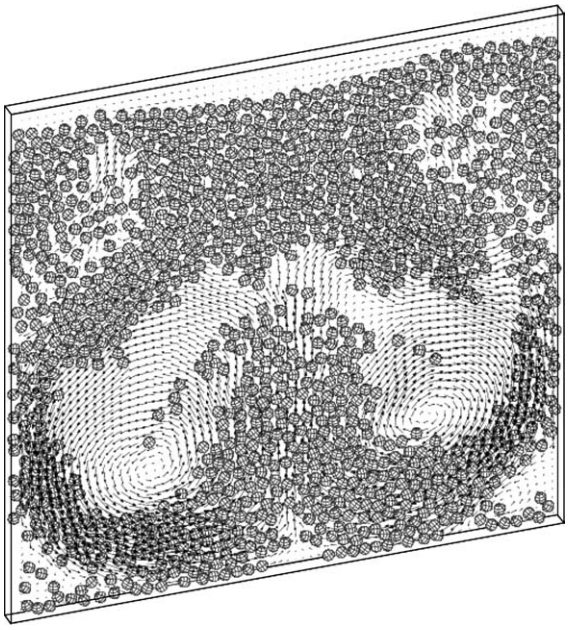

Fig. 27. Particle positions at $t = 8$ s.

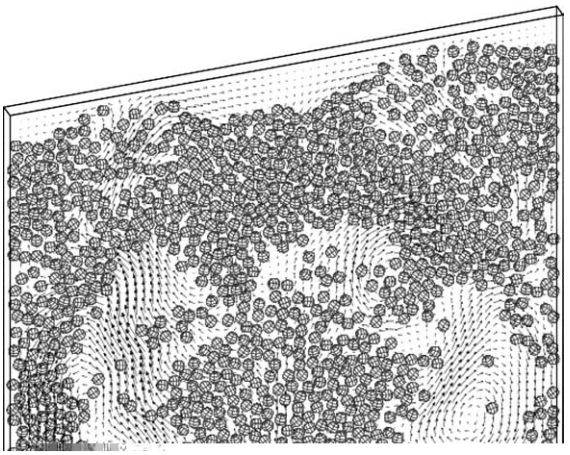

Fig. 28. Particle positions at $t = 9$ s.

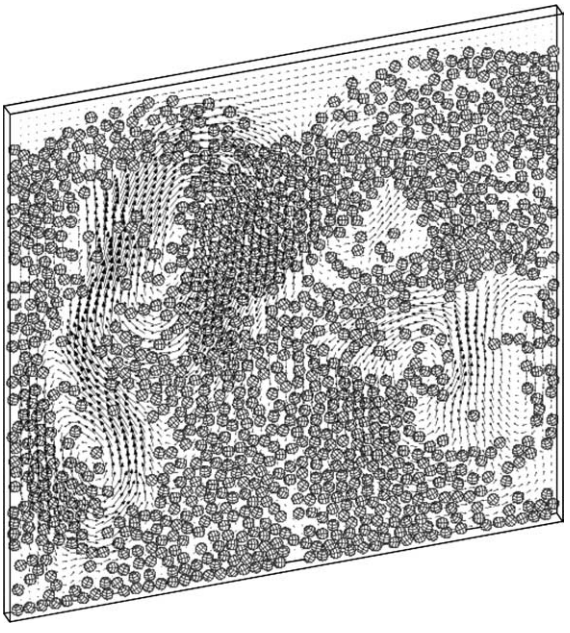

Fig. 29. Particle positions at $t = 10$ s.

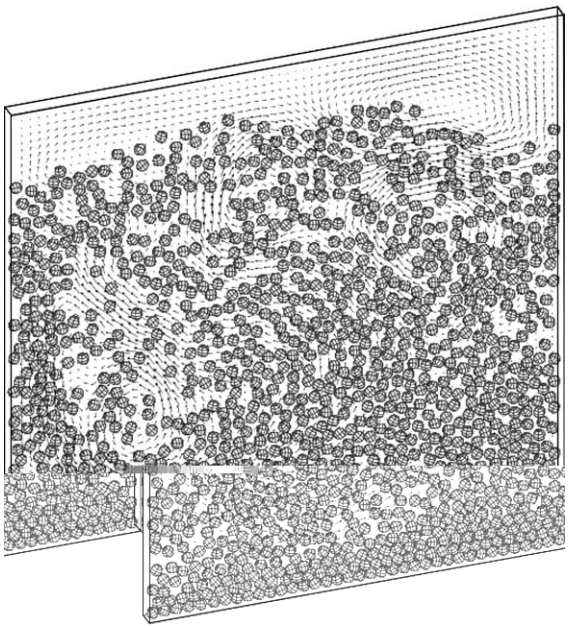

Fig. 30. Particle positions at $t = 15$ s.

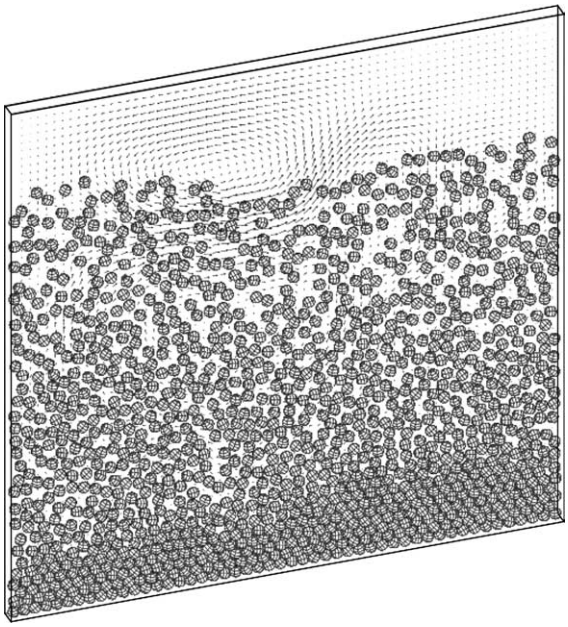

Fig. 31. Particle positions at $t = 20$ s.

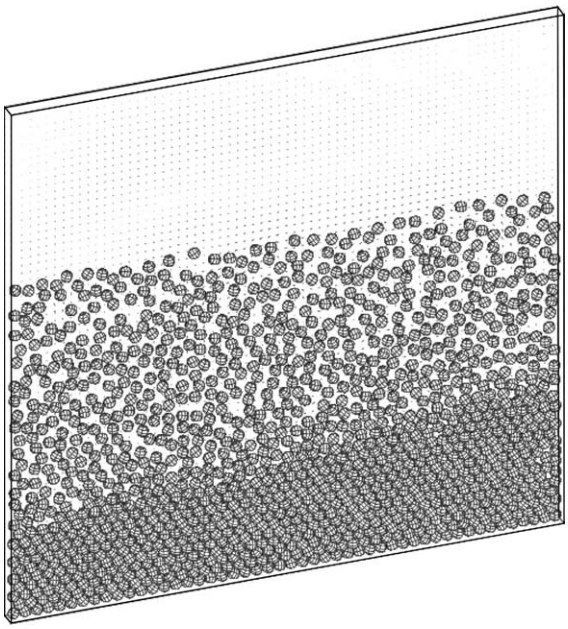

Fig. 32. Particle positions at $t = 30$ s.

section of the enclosure. The upward motion of the particles still persists at 15 s and it takes considerably longer for the upper part of the enclosure to become clear of particles. This type of Rayleigh–Taylor motion is very similar to the two-dimensional motion that was observed in the computations of Glowinski et al. [13].

5. Conclusions

An efficient three-dimensional computational method, *Proteus*, which is based on the lattice Boltzmann method the immersed boundary method and the direct forcing scheme, has been developed for use with large groups of particles. This method combines desired characteristics of the three computational methods to achieve higher efficiency and accuracy. *Proteus* computes the force term directly and does not require the use of any other coefficients as additional parameters. Compared with the conventional LBM, the new method provides a smooth computational boundary at comparable computational cost. While it has the same order of accuracy as the LBM, *Proteus* is capable of achieving results at higher Reynolds numbers and, if needed, to easily enforce the rigid body motion in the interior of the particles. *Proteus* is also easier and more efficient to be used when the particles do not have a simple shape.

The new method has been validated by comparison with results from the simulations of the motion of single spheres settling in an enclosure with analytical and experimental results that were derived in the past. The method has been successfully applied to the sedimentation problem of an arrangement of 1232 spheres in a narrow enclosure under two different boundary conditions. The results show that the new computational method yields accurate predictions for large groups of interacting particles and that it is a very efficient and promising technique to be applied to particle–fluid interaction problems.

Acknowledgement

This research was partly supported by two grants from the USGS and the NOAA to the Tulane-Xavier Center for Bioenvironmental Research. Computational resources were provided by the Tulane/Xavier Livingston Millennium Center for Computational Studies. Logistic assistance has been provided by the South-central Regional Center of NIGEC, which is supported by the BER division of the DOE.

References

[1] F. Abraham, Functional dependence of drag coefficient of a sphere on Reynolds number, Phys. Fluids 13 (1970) 2194.
[2] C.K. Aidun, Y.-N. Lu, E.-J. Ding, Direct analysis of particulate suspensions with inertia using the discrete Boltzmann equation, J. Fluid Mech. 373 (1998) 287.
[3] T. Bohlin, Terminal velocities of solid spheres in cylindrical enclosures, Transactions of the Royal Institute of Technology, Stockholm, 1960, Report# 155.
[4] J.F. Brady, G. Bossis, Stokesian dynamics, Ann. Rev. Fluid. Mech. 20 (1988) 111–157.
[5] S. Chen, G.D. Doolen, Lattice Boltzmann method for fluid flows, Annu. Rev. Fluid Mech. 30 (1998) 329–364.
[6] H. Faxen, Der Widerstand gegen die Bewegung einer starren Kugel in einer zum den Flussigkeit, die zwischen zwei parallelen Ebenen Winden eingeschlossen ist, Annalen der Physik 68 (1922) 89–119.
[7] J. Feng, H.H. Hu, D.D. Joseph, Direct simulation of initial value problems for the motion of solid bodies in a Newtonian fluid. Part 1: sedimentation, J. Fluid Mech. 261 (1994) 95–134.
[8] J. Feng, H.H. Hu, D.D. Joseph, Direct simulation of initial value problems for the motion of solid bodies in a Newtonian fluid. Part 2: Couette and Poiseuille flows, J. Fluid Mech. 277 (1994) 271–301.
[9] Z.-G. Feng, E.E. Michaelides, An immersed boundary method combined with lattice Boltzmann method for solving fluid and particles interaction problems, J. Comput. Phys. 195 (2004) 602–628.
[10] Z.G. Feng, E.E. Michaelides, Hydrodynamic force on spheres in cylindrical and prismatic enclosures, Int. J. Multiphase Flow 28 (2002) 479–496.
[11] Z.G. Feng, E.E. Michaelides, Interparticle forces and lift on a particle attached to a solid boundary in suspension flow, Phys. Fluids 14 (2002) 49–60.
[12] A.L. Fogelson, C.S. Peskin, A fast numerical method for solving the three-dimensional Stokes equation in the presence of suspended particles, J. Comput. Phys. 79 (1988) 50–69.
[13] R. Glowinski, T.-W. Pan, T.I. Hesla, D.D. Joseph, A distributed Lagrange multiplier/fictitious domain method for particulate flows, Int. J. Multiphase Flow 25 (1999) 755–794.
[14] R. Glowinski, T.-W. Pan, T.I. Hesla, D.D. Joseph, J. Periaux, A fictitious domain approach to the direct numerical simulation of incompressible viscous flow past moving rigid bodies: application to particulate flow, J. Comput. Phys. 169 (2001) 363–426.

[27] T. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, Y.-J. Jan, A front-tracking method for the computations of multiphase flow, J. Comput. Phys. 169 (2001) 708–759.
[28] Z. Zhang, A. Prosperetti, A method for particle simulation, J. Appl. Mech.-Trans. ASME 70 (2003) 64–74.