Chemical Engineering Journal 152 (2009) 514–529

Contents lists available at ScienceDirect

Chemical Engineering Journal

journal homepage: www.elsevier.com/locate/cej

Three-dimensional discrete particle model for gas–solid fluidized beds on unstructured mesh

C.L. Wu [a,*], A.S. Berrouk [b], K. Nandakumar [b]

[a] Engineering College, Guangdong Ocean University, Zhanjiang 524088, China
[b] Department of Chemical Engineering, The Petroleum Institute, P.O. Box 2533, Abu Dhabi, United Arab Emirates

ARTICLE INFO

Article history:
Received 3 February 2009
Received in revised form 9 May 2009
Accepted 17 May 2009

Keywords:
Fluidization
Multiphase flow
Computation efficiency
Hard-sphere model
Unstructured mesh

ABSTRACT

A robust three-dimensional discrete particle model (3D DPM) for unstructured meshes is presented to model the gas–solid flows in fluidized beds. The finite volume method is used to discretize the governing equations of the gas phase. Inter-particle interactions are taken into account through the hard-sphere approach. The numerical models for the solid phase were implemented as a separate module in FLUENT by overcoming some limitations of the user-defined function approach. A multifaceted numerical strategy was developed to enhance the computational efficiency. Simulation results demonstrate the ability of the model to capture many important characteristics such as bubbling, spouting, particle clustering and core–annulus flow structures in gas–solid systems at specified operating conditions.

1. Introduction

Gas–solid fluidized beds are widely used in the chemical and petrochemical industries for a large variety of processes, such as catalytic cracking, drying, coating, coal gasification and combustion. The ongoing research efforts to develop numerical models for such complex systems are significant. Due to the multi-scale character of these systems, several models at different scale levels have been formulated [1–4]. At the macroscopic level, mixture or two-fluid models (TFMs) need closure laws to describe the two-phase interaction and the particulate matter rheology. Although these macroscopic models are the most tractable to simulate actual gas–solid systems that could help their optimization, they are known to be silent about microstructure and they have never acknowledged the very important microstructural properties that are induced by particle–fluid interaction. This has motivated the recourse to finer numerical descriptions at the microscopic scale. Indeed, at that level of description, direct numerical simulation (DNS) has been used to fully resolve the gas flow details around the particles. DNS models based on Lattice Boltzmann method, immersed boundary method and fictitious domain or arbitrary Lagrangian–Eulerian methods do not require closure relations. However, they are computationally intensive and resource demanding when applied to dense gas–particle systems of practical relevance, which are three-dimensional and contain a large number of particles. This has made DNS models unwieldy to apply for practical engineering problems. Notwithstanding this limitation, however, the DNS results are of great value to understand the physical mechanisms through which fluid and particle interact and help derive more accurate closure relations [5,6].

Between these two levels of numerical description, a mesoscopic approach for particulate flows has been developed under the name of discrete particle model (DPM). In this approach, the particle motion is described in a Lagrangian framework by directly solving the Newtonian kinetic equations of each individual particle. The particulate-phase constitutive relations are not required because the particle–particle interactions are modeled through a two-variant collision strategy; the hard-sphere variant or the soft-sphere variant.

The DPM has become a versatile tool since the pioneering works that focused on its development and validation [1,2,7,8]. It has been widely used over the last decade to study the complicated flow behaviors in gas–solid fluidized beds. The numerical results obtained have demonstrated the powerful capabilities of the method [9–13]. Although the gas flow details around particles are not well determined, the particle motion is resolved at such a particle scale that many important features related to the particle motion in fluidized beds are reasonably captured. For instance, several investigations based on DPMs have helped to understand that the formation of heterogeneous structures in fluidized beds is attributed to the combination of energy dissipation due to inter-particle interactions and the strong dependence of the inter-phase drag force on the void fraction [1,8,13–16]. These closure relations

* Corresponding author. Tel.: +86 7592383720/13828261515.
E-mail address: chunliangwu@gmail.com (C.L. Wu).

1385-8947/$ – see front matter.
doi:10.1016/j.cej.2009.05.024

for the inter-phase drag force are deduced from experimental data, theoretical or numerical results of micro-scale models [17–19].

The state-of-the-art review on discrete particle modeling was recently presented by Deen et al. [3]. Although a lot of progress has been made on the numerical aspects of DPM, many studies were based on the two-dimensional (2D) model and only structured regular grids were adopted to discretize the computational domain. A 2D model that restricts the degree of freedom of two naturally interacting particles to be scattered in a natural three-dimensional domain to be confined to a 2D space is restrictive to be of any practical relevance. Despite the relatively high computational cost of 3D DPM compared to 2D DPM, it appears to be the only realistic way to simulate many industrial processes taking place in complex geometries. From a theoretical point of view, the 2D model for instance cannot characterize the radial hydrodynamics in cylindrical circulating fluidized bed (CFB) risers because the flow in a riser is not axis-symmetric in most of the cases. Therefore, 3D DPM implemented in conjunction with unstructured grid should be developed and extended to incorporate necessary details to model real-world chemical engineering problems. Compared to structured grids, unstructured grids have the natural flexibility to mesh complex geometries and they are widely used in numerical simulations of single-phase flow [20] and two-phase flow with free surface [21]. They have also been applied to simulate gas–solid flows in fluidized beds based on TFM [22]. It is almost unavoidable to use unstructured grids to mesh complex solution domains characterizing many real-life fluidization systems. These systems usually contain many complex internals such as immersed tubes, draft tubes or baffles that make the use of the structured grid ineffective. However, the use of unstructured grids in conjunction with DPM brings about some numerical difficulties regarding the void fraction calculation for arbitrary particle locations and cell shapes. These issues should be well addressed in order to do not discount the advantages brought by the use of unstructured grids.

The high computational cost linked to 3D DPM has limited its use to applications in a laboratory scale with the number of particles typically less than 100,000. This shortcoming can be overcome by designing a computationally efficient multifaceted numerical strategy. It is deemed that the model accuracy and its implementation efficiency depend strongly but not solely on: (1) 3D void fraction computation, (2) inter-phase coupling, and (3) collision event handling through proper data structures.

The objective of this work is to develop a generic and efficient hard-sphere discrete particle model for gas–solid flows in arbitrary 3D domains. A methodology of implementing the model in the commercial CFD code FLUENT is described along with an efficient many-sided numerical strategy. The paper is organized into the following sections. First, a brief discussion of the equations governing both gas and particles dynamics is presented followed by a section on how the issues surrounding their implementation in FLUENT are resolved. Subsequently, the multifaceted numerical strategy is detailed. Finally, the 3D DPM is validated on a number of problems involving gas–solid fluidized beds for which experimental data exist.

2. Governing equations of the gas–particle system

The governing equations of the gas phase mainly follow the volume-averaged form of the two-fluid model (Model A) presented by Gidaspow [17]. This set of equations has been widely applied in the discrete particle/element studies mentioned in the introduction. The particles' motion obeys Newtonian second law of motion and particle collisions are described by a three-parameter hard-sphere model [23,24]. Interactions between the two phases are accounted for through the momentum exchange term.

2.1. Gas-phase hydrodynamics

In accordance with the two-fluid model, the conservation of mass is balanced by the convective mass fluxes without inter-phase mass transfer:

$$\frac{\partial}{\partial t}(\varepsilon\rho_g) + \nabla\cdot(\varepsilon\rho_g\boldsymbol{u}_g) = 0 \tag{1}$$

where ρ_g is the density of the gas phase, ε is the void fraction and $\mathbf{u}_g$ is the gas velocity.

The momentum balance equations are similar to the single-phase momentum equations:

$$\frac{\partial}{\partial t}(\varepsilon\rho_g\boldsymbol{u}_g) + \nabla\cdot(\varepsilon\rho_g\boldsymbol{u}_g\boldsymbol{u}_g) = -\varepsilon\nabla p + \nabla\cdot(\varepsilon\bar{\bar{\boldsymbol{\tau}}}_g) + \mathbf{S}_g + \varepsilon\rho_g\mathbf{g} \tag{2}$$

where p is the static pressure and $\bar{\bar{\boldsymbol{\tau}}}_g$ the stress tensor. The source term $\mathbf{S}_g$ due to particle–fluid drag interaction was suggested by Goldschmidt et al. [23]:

$$\boldsymbol{S}_g = -\frac{1}{V}\int_V \sum_{k=1}^{N_P} \frac{V_{p,k}\beta}{1-\varepsilon}(\boldsymbol{u}_g - \boldsymbol{u}_{p,k})\delta(\boldsymbol{x}-\boldsymbol{x}_{p,k})dV \tag{3}$$

where V is the integral volume, V_p is the particle volume, N_p is the particle number, $\mathbf{u}_p$ is the particle velocity and the δ function ensures that the reaction force acts as a point force at the position of the particle. β is the drag coefficient, which is a function of the particle Reynolds number, Re_p and the void fraction, ε.

2.2. Dispersed phase hydrodynamics

Since the gas inertia is much smaller than that of the solid particles, only the drag and the pressure gradient forces are taken into account in the particle momentum equation, which is written in a Lagrangian framework and reads as follows:

$$m_p\frac{d\boldsymbol{u}_p}{dt} = -V_p\nabla p + \frac{V_p\beta}{1-\varepsilon}(\boldsymbol{u}_g - \boldsymbol{u}_p) + m_p\mathbf{g} \tag{4}$$

where m is the particle mass.

In the hard-sphere approach, collisions among dispersed particles are assumed binary and quasi-instantaneous and particle contact occurs at one point. During a collision only impulse forces are considered. The relative velocity $\mathbf{v}_{12}$ at the contact point between two particles with velocities $\mathbf{v}_1$ and $\mathbf{v}_2$ is defined as

$$\mathbf{v}_{12} = (\mathbf{v}_1 - \mathbf{v}_2) - (r_1\boldsymbol{\omega}_1 + r_2\boldsymbol{\omega}_2)\times\mathbf{n} \tag{5}$$

where r_1 and r_2 are the radii, and $\boldsymbol{\omega}_1$ and $\boldsymbol{\omega}_2$ the angular velocities.

The normal and tangential unit vectors that define the collision coordinate system are

$$\mathbf{n} = \frac{\mathbf{x}_1 - \mathbf{x}_2}{|\mathbf{x}_1 - \mathbf{x}_2|} \tag{6}$$

$$\mathbf{t} = \frac{\mathbf{v}_{12}^0 - (\mathbf{G}_{12}^0\cdot\mathbf{n})\mathbf{n}}{\left|\mathbf{v}_{12}^0 - (\mathbf{G}_{12}^0\cdot\mathbf{n})\mathbf{n}\right|} \tag{7}$$

where the superscript 0 denotes conditions just before collision and $\mathbf{G}_{12}$ is the relative velocity of particle centroids. The equations of motions of the two colliding particles are given in the following:

$$\begin{aligned} &m_1(\mathbf{v}_1 - \mathbf{v}_1^0) = \mathbf{J} \\ &m_2(\mathbf{v}_2 - \mathbf{v}_2^0) = -\mathbf{J} \\ &\frac{I_1}{r_1}(\boldsymbol{\omega}_1 - \boldsymbol{\omega}_1^0) = \mathbf{J}\times\mathbf{n} \\ &\frac{I_2}{r_2}(\boldsymbol{\omega}_2 - \boldsymbol{\omega}_2^0) = \mathbf{J}\times\mathbf{n} \end{aligned} \tag{8}$$

where I_1 and I_2 are the moments of inertia and $\mathbf{J}$ is the impulse vector.

By introducing the first parameter: the coefficient of normal restitution e_n, we can get the normal component of the impulse vector:

$$J_n = -m_1 m_2 (1 + e_n) \frac{\mathbf{v}_{12}^0 \cdot \mathbf{n}}{(m_1 + m_2)} \tag{9}$$

The second and the third collision parameters are the coefficient of friction μ_f and the coefficient of tangential restitution e_t. These two collision parameters represent two kinds of collisions, namely sticking and sliding in the tangential impact process. The tangential component of impulse is given by

$$J_{t,sliding} = -\mu_f J_n \quad \text{for } \mu_f < \frac{2m_1 m_2 (1 + e_t)\mathbf{v}_{12}^0 \cdot \mathbf{t}}{7(m_1 + m_2) J_n}$$

$$J_{t,sticking} = -(1 + e_t)\frac{2m_1 m_2 \mathbf{v}_{12}^0 \cdot \mathbf{t}}{7(m_1 + m_2)} \quad \text{for } \mu_f \geq \frac{2m_1 m_2 (1 + e_t)\mathbf{v}_{12}^0 \cdot \mathbf{t}}{7(m_1 + m_2) J_n} \tag{10}$$

With the total impulse vector known the post-collision velocities can be computed from Eq. (8).

3. Model implementation in FLUENT

The commercial CFD software FLUENT is a state-of-the-art computer program for modeling fluid flow and heat transfer in complex geometries. Several mathematical models for two-phase flow are already coded in FLUENT, such as the mixture model, the two-fluid Eulerian model, the VOF and the discrete phase model. Only the latter model can be used to simulate dispersed gas–solid flows in the Lagrangian framework. However, it is limited to sufficiently dilute cases where particle–particle interactions and the effects of the particle volume fraction on the gas phase are negligible [25].

In this section, we detail an efficient methodology to implement the discrete particle model in FLUENT as described in Section 2. In DPMs, due to the weak gas-phase inertia compared to that of the dispersed phase, the complex gas–particle system is often divided in two subsystems or processes that are solved separately at each simulation time step. During the first step, the two-phase interactions are taken into account while the particles are assumed fixed in space. Thus, only the momentum exchange between the two phases is accounted for in this first step. The gas-phase governing equations are solved using a finite volume/difference method together with the particle momentum equation. The system of two-phase equations is solved using either an explicitly segregated scheme or an implicitly coupled algorithm. The time step should be selected at least one order smaller than the particle response time to capture the two-phase interaction. In the second step, all possible collisions between particles are detected and the collision dynamics is computed for each occurring collision. During this step, all collision events are handled one by one in a chronological order.

In a previous work, we developed a in-house code of the hard-sphere DPM on 2D unstructured mesh [24]. However, in this work, we choose FLUENT as our DPM platform. This is because it uses the finite volume method to solve the transport equations and provides complete mesh flexibility supporting both structured and unstructured meshes that can be generated for complex geometries, which are often encountered in fluidization system, such as cylindrical riser and conical spouted fluidized bed. Also the algebraic multi-grid (AMG) solver used in FLUENT is very robust and several pressure–velocity coupling algorithms are offered as options. In addition, many in-line models coded in FLUENT has been validated and successfully applied to solve fluid flow and heat transfer problem in industry, which supports a future possibility to further integrate these models into the present DPM code as an extension. The ability of the user-defined functions (UDFs) supported in FLUENT allows the user to enhance the standard features of the code, such as customized boundary conditions, material properties and source/sink terms in the transport equation. These FLUENT features enable us to integrate the hard-sphere model into its basic fluid flow solver. In the following, we show first the limitation of a normal UDF development of the DPM, then how to overcome this limitation by re-arranging the governing equations of the gas phase.

3.1. Normal UDF implementation

It is not applicable to develop the DPM code grounding upon the standard multiphase flow models (such as the discrete phase model or the Eulerian model) in FLUENT, since the code details of these models are hidden from the user and it is difficult to disable some unnecessary features linked to these models. The developed code is supposed to be fully customized by the user, especially regarding the hydrodynamics of the dispersed particles. We start from the basic solver for the incompressible single-phase flow, governed by the following set of equations:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}_g) = 0 \tag{11}$$

$$\frac{\partial}{\partial t}(\rho \mathbf{u}_g) + \nabla \cdot (\rho \mathbf{u}_g \mathbf{u}_g) = -\nabla p + \nabla \cdot (\bar{\bar{\tau}}_g) + \mathbf{S}_m + \rho \mathbf{g} \tag{12}$$

As said above, the DPM procedure consists of the resolution of the gas and particle governing equations (Eqs. (1), (2), and (4)) at the beginning of each time step followed by the particle collision dynamics. Before solving the two-phase equations, the void fraction is calculated according to the particle positions and the cell geometry of the finite volume meshes. The basic steps of the model implementation are shown in Fig. 1. By comparing the single-phase governing equations to those of the gas phase in DPM, one finds a straightforward way to incorporate DPM into FLUENT. It consists of introducing one UDF to compute the density property of the gas phase and another one to calculate the momentum source terms due to two-phase coupling, as follows:

$$\rho = \rho_g \varepsilon \tag{13}$$

$$\mathbf{S}_m = \mathbf{S}_g + \varepsilon_p \nabla p \tag{14}$$

where $\varepsilon_p = 1 - \varepsilon$ is the particle volume fraction. User-defined memories for each cell are used to store the calculated void fraction and the source value.

After a comprehensive numerical testing, however, we found that the above UDF approach cannot correctly capture the bubbling characteristic of the Geldart B particles in fluidized beds. In addition, very slow convergence of the solution was noticed when the SIMPLE algorithm was used in conjunction with the AMG solver. Through a careful examination of the code, it was found that the above problem is due to the incorrect pressure predictions in the bed that might be incurred by the momentum source term evaluation. The second part in the right hand side of Eq. (14) represents the particle reaction force of the pressure gradient exerted on the gas phase, which is treated as an explicit cell source term and is not discretized as the main coefficients in the pressure correction equation. Because of the strong coupling description of the pressure and velocity in the SIMPLE algorithm for incompressible gas flow, the explicit pressure gradient source term in the momentum equations results in an adverse convergence of the pressure correction process. As a conclusion, the explicit calculation of the pressure gradient force fails to correctly predict the two-phase interaction. This problem can be overcome by considering an improved UDF implementation as shown below.

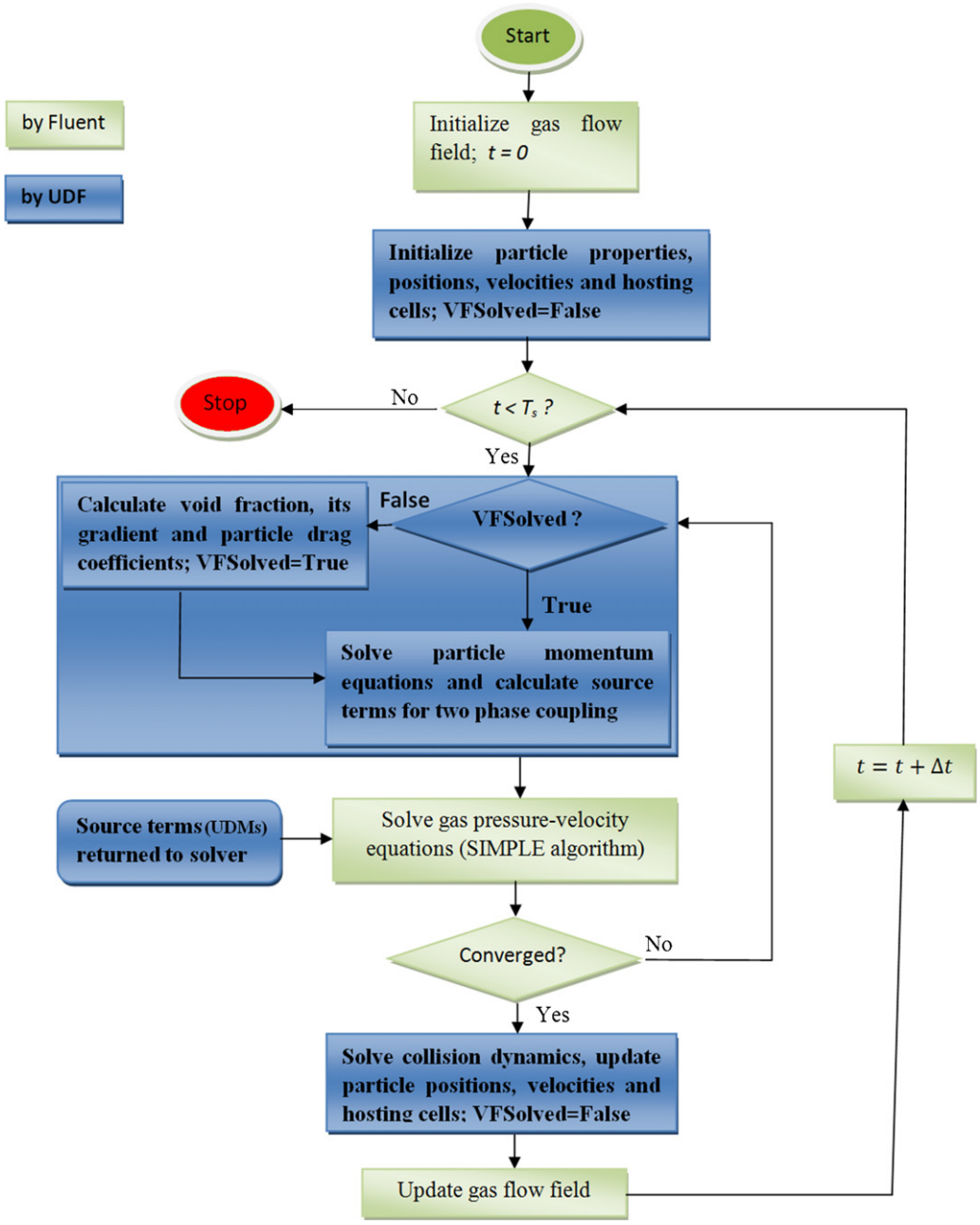

Fig. 1. Flowchart of the DPM for FLUENT. T_s is the simulation time given by the user.

3.2. Improved UDF implementation

To overcome the limitation of the normal UDF approach, we firstly re-organize the continuum equation (Eq. (1)) of the gas phase as follows:

$$\frac{\partial \rho_g}{\partial t} + \nabla \cdot (\rho_g \mathbf{u}_g) + \frac{\rho_g}{\varepsilon}\left(\frac{\partial \varepsilon}{\partial t} + \mathbf{u}_g \cdot \nabla \varepsilon\right) = 0 \tag{15}$$

or

$$\frac{\partial \rho_g}{\partial t} + \nabla \cdot (\rho_g \mathbf{u}_g) = S_c \tag{16}$$

with

$$S_c = -\frac{\rho_g}{\varepsilon}\left(\frac{\partial \varepsilon}{\partial t} + \mathbf{u}_g \cdot \nabla \varepsilon\right) \tag{17}$$

If the void fraction is considered as a scalar property of the gas phase, Eq. (16) demonstrates that the flow flux in unit gas volume equals the relative variation rate of the void fraction if we consider the fact that the gas density is constant. The terms in the left hand side of Eq. (16) represent the net mass outflow in unit gas volume. The term in the right hand side can be considered as a mass source term due to the incompressible constraint of the gas phase and the volume fraction variation caused by the particle motion.

The momentum equations of the gas phase are re-arranged as follows:

$$\varepsilon \frac{\partial}{\partial t}(\rho_g \mathbf{u}_g) + \varepsilon \nabla \cdot (\rho_g \mathbf{u}_g \mathbf{u}_g) + \rho_g \mathbf{u}_g \left(\frac{\partial \varepsilon}{\partial t} + \mathbf{u}_g \cdot \nabla \varepsilon\right) = -\varepsilon \nabla p + \nabla \cdot (\varepsilon \bar{\bar{\tau}}_g) + \mathbf{S}_g + \varepsilon \rho_g \mathbf{g} \tag{18}$$

or

$$\frac{\partial}{\partial t}(\rho_g \mathbf{u}_g) + \nabla \cdot (\rho_g \mathbf{u}_g \mathbf{u}_g) = -\nabla p + \nabla \cdot (\bar{\bar{\tau}}_g) + \rho_g \mathbf{g} + \mathbf{S}_m \tag{19}$$

where

$$\mathbf{S}_m = \frac{\mathbf{S}_g}{\varepsilon} - \frac{\mathbf{u}_g \rho_g}{\varepsilon}\left(\frac{\partial \varepsilon}{\partial t} + \mathbf{u}_g \cdot \nabla \varepsilon\right) + \frac{\bar{\bar{\tau}}_g \cdot \nabla \varepsilon}{\varepsilon}$$

or

$$\mathbf{S}_m = S_c \mathbf{u}_g + \frac{1}{\varepsilon}(\mathbf{S}_g + \bar{\bar{\tau}}_g \cdot \nabla \varepsilon) \tag{20}$$

The first part in the above source term as shown in Eq. (20) is the gas momentum increase due to the variation of the void fraction. Since the momentum variation rate shown in the left hand side of Eq. (19) is evaluated in unit gas volume, the momentum exchange rate due to gas–solid interactions should also be evaluated in unit gas volume, just as shown in the second part of the source equation (Eq. (20)). The third part is an additional viscous force due to the non-uniform distribution of void fraction, which is much smaller compared to the other two parts according to our numerical simulations. Thus this term is neglected in the final implementation of the code.

By comparing the deduced governing equations (Eqs. (16) and (19)) for the gas phase to those (Eqs. (11) and (12)) for single-phase flow, it can be seen that their incorporation into FLUENT is straightforward. Indeed, only the source terms given above are defined as UDFs for the gas continuity and momentum equations. In each time step and before calculating these source terms, the gradient of the void fraction is calculated altogether with its local variation rate. The momentum source terms are linearized as follows:

$$\mathbf{S}_m = \mathbf{S}_a + B\mathbf{u}_g \tag{21}$$

$$B = S_c - \frac{1}{V}\int_V \sum_{k=1}^{N_p} \frac{\beta V_{p,k}}{\varepsilon(1-\varepsilon)} \delta(\mathbf{x} - \mathbf{x}_{p,k}) dV \tag{22}$$

$$\mathbf{S}_a = \frac{1}{V}\int_V \sum_{k=1}^{N_p} \frac{\beta V_{p,k} \mathbf{u}_{p,k}}{\varepsilon(1-\varepsilon)} \delta(\mathbf{x} - \mathbf{x}_{p,k}) dV \tag{23}$$

The above approach exhibits a good convergence performance of the FLUENT AMG solver when the SIMPLE algorithm is applied. In addition, the implemented model was found to be able to capture many important phenomena in gas–solid fluidized bed, as shown later. The improved approach brought about the following advantages: (1) no explicit term related to the pressure gradient is present in the pressure correction equations and only the velocities and the void fraction are involved. Thus the pressure can be correctly predicted. (2) The source terms in the gas momentum equations are linearized and the coefficients are automatically absorbed into the discretized equations as principle coefficients, which greatly improves the numerical stability. (3) The resulting source terms are more related to the void fraction, i.e. to its local variation rate and gradient and they can be calculated very precisely based on the particle position. As shown in Fig. 1, when solving the particle momentum equations, the source terms should be calculated and stored in user-defined memories (UDMs) to avoid additional loop, since the user-defined source functions are called by the solver at the cell level.

4. Multifaceted numerical strategy

To ensure an efficient implementation of the discrete particle model used to simulate dense gas–particle systems on unstructured grids, a many-sided numerical strategy is designed. This strategy is based on: (i) an accurate calculation of the void fraction, (ii) a strong implicit coupling of the two-phase interaction and (iii) an adequate data structure designed to obtain a complexity $O(1)$ time per collision regarding the event handling of particle collisions. The corresponding codes are also plugged into FLUENT as additional modules.

4.1. Void fraction calculation

The void fraction should be estimated in a prudent way in order to support the unstructured mesh feature of FLUENT. This constitutes a challenge when particles are not fully contained in one cell that can have any shape (wedged, tetrahedral, or hexahedral) and/or can intersect particles through any of its boundaries (node, edge or face). Indeed, when the centre of a particle locates on an edge shared by two 2D cells or on a face shared by two 3D cells, there will be a considerable error in the calculated porosity if the volume of this particle is not accurately shared between these two cells. Since the porosity plays a very important role in the local mass and momentum balance of the gas phase and the drag coefficient are found to be strongly dependent on it, the said errors should be avoided as far as possible.

An analytical method to calculate the void fraction for general 3D unstructured meshes is proposed herein. It is developed for cells of different shapes (wedged, tetrahedral, or hexahedral) and for different particle–cell intersections (node, edge or face). Four cases are identified for which the volume V_n is accurately computed as it is portrayed in Fig. 2. These four cases are deemed to cover all possible configurations for the cell–particle intersection. Obviously if V_n is known then we get $V_0 = (4/3)\pi r_p^3 - \sum V_n$. Herein V_0 and V_n ($n = 1$, 2) are the volume of the cell C_0 hosting the centre of the particle and the ones of the other cells C_n that share the particle, respectively.

Case 1 : the particle is wholly contained in C_0, then $V_n = 0$.
Case 2 : a segment of the volume of the sphere is shared by cell C_n totally through one of its faces. The volume of this segment can be calculated by $V_n = \pi r_n^2 (r_p - (1/3)r_n)$, where $r_n = r_p - h$

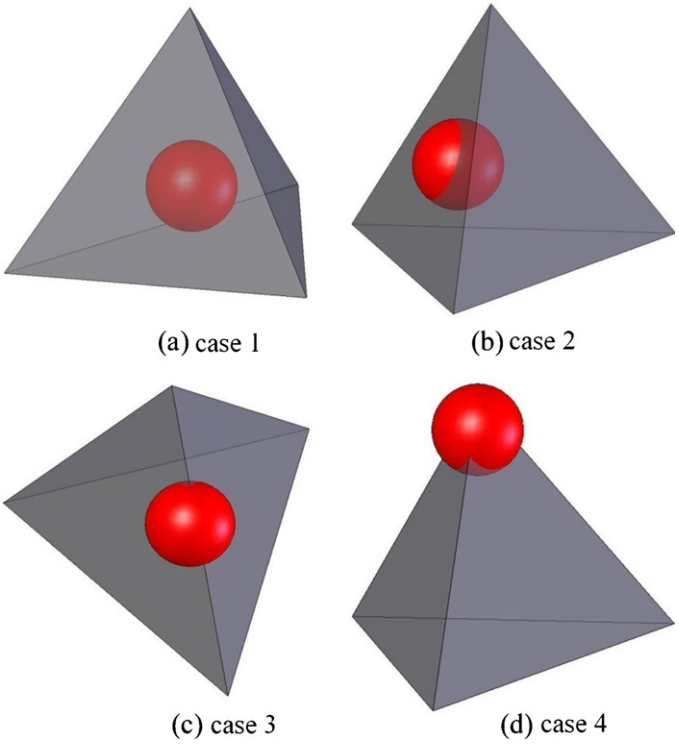

Fig. 2. Particle positions in 3D cells.

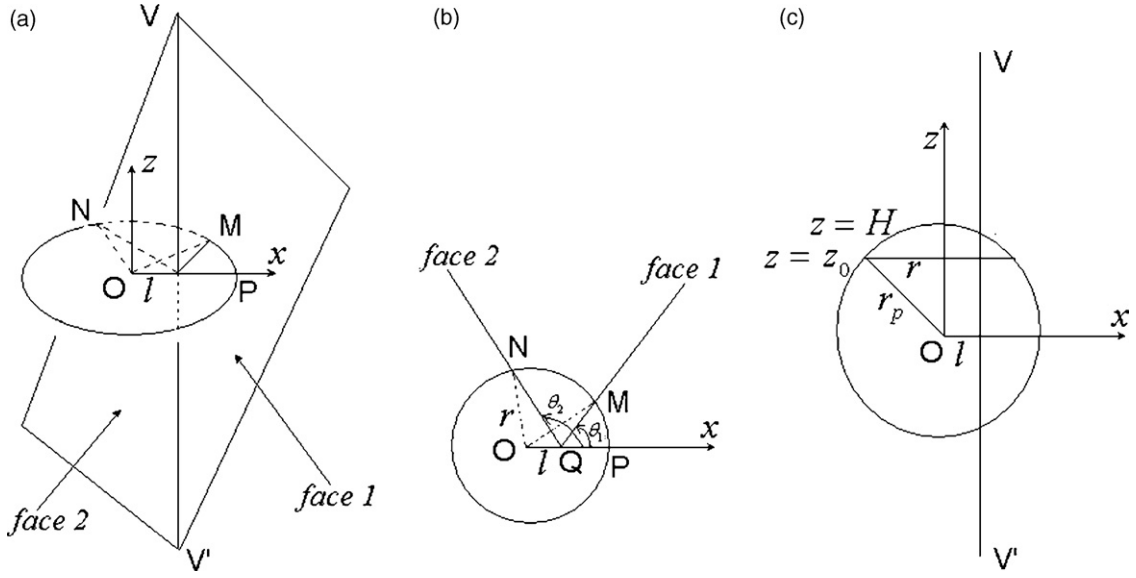

Fig. 3. Accurate calculation of porosity on 3D unstructured mesh (case 3): (a) perspective view, (b) plane at $z=0$ and (c) x–z plane.

and h is the distance from the particle centre to the face splitting the particle sphere. In this case, the particle may be split into several parts by more than one face if the angle between the two adjacent faces of a cell is sharp.

Case 3 : one edge (or more than one) of the cell C_0 passes through the sphere, so the particle is split into several parts by all the faces sharing this edge. To analyze this case, we specify this edge as VV', the particle centre O as the origin of the local cylindrical coordinate system, the line parallel to VV' as the z-axis, and the line normal to the edge VV' as the x-axis (see Fig. 3a). The plane at $z=0$ is shown in Fig. 3b and the x–z plane in Fig. 3c. The volume of the sphere cut by the two faces (face 1 and face 2) of the cell C_n can be calculated by

$$V_n(l,\theta_1,\theta_2) = V(l,\theta_2) - V(l,\theta_1) \tag{24}$$

$$V(l,\theta) = 2\int_0^H S(l,\theta,z)dz \quad \left(0 \le \theta \le \frac{\pi}{2}, 0 < l < r_p\right)$$

$$V(l,\theta) = \pi s^2\left(r_p - \frac{s}{3}\right) - 2\int_0^H S(l,\pi-\theta,z)dz \quad \left(\frac{\pi}{2} < \theta \le \pi, 0 < l < r_p\right)$$

Here the integrand S is defined as

$$S(l,\theta,z) = \frac{1}{2}r(z)^2\left[\theta - \arcsin\left(\frac{l\sin\theta}{r(z)}\right)\right] - \frac{1}{2}l(\sqrt{r(z)^2 - l^2\sin^2\theta} - l\cos\theta)\sin\theta$$

$r(z) = \sqrt{r_p^2 - z^2}$ and $s = r_p - l\sin\theta$. The upper limit of the above integrals is $H = \sqrt{r_p^2 - l^2}$ $(0 < l < r_p)$.

In the above definition of V_n, l is the distance from the particle centre to VV', θ_1 and θ_2 are the angles from the x-axis to face 1 and face 2, respectively.

$S(l,\theta,z_0)$ represents the area of the pseudo sector VMP at the height $z=z_0$. Integration yields

$$V(l,\theta) = 2\int_0^H S(l,\theta,z)dz = I_0 + \frac{l\sin\theta}{3}(r_p^2 I_1 + I_2) \tag{25}$$

$$I_0 = \frac{1}{2}l\sin\theta\int_0^H (l\cos\theta - \sqrt{r(z)^2 - l^2\sin^2\theta})dz$$
$$= \frac{1}{2}l^2 H\sin\theta\cos\theta - \frac{1}{2}l\sin\theta(r_p^2 - l^2\sin^2\theta)\arcsin\left(\sqrt{\frac{r_p^2 - l^2}{r_p^2 - l^2\sin^2\theta}}\right)$$

$$I_1 = \int_l^{r_p}\frac{1}{r^2}\sqrt{\frac{r_p^2 - r^2}{r^2 - l^2\sin^2\theta}}dr^2$$
$$= \frac{2r_p}{l\sin\theta}\arctan\left(\frac{H\tan\theta}{r_p}\right) - 2\arctan\left(\frac{H}{l\cos\theta}\right)$$

$$I_2 = \frac{1}{2}\int_l^{r_p}\sqrt{\frac{r_p^2 - r^2}{r^2 - l^2\sin^2\theta}}dr^2$$
$$= \frac{1}{2}(r_p^2 - l^2\sin^2\theta)\arctan\left(\frac{H}{l\cos\theta}\right) - \frac{1}{2}Hl\cos\theta$$

Once the length l and the angle θ are calculated according to the positions of the particle and the cell faces, the above integrals yield the sphere volume encompassed by the x–z plane and the face. All the faces sharing the edge VV' are identified first and the angle from the local x-axis to each face are calculated in a loop to avoid repetitive computations. Then the volume of a sphere segment cut by any two faces can be calculated using Eqs. (24) and (25). If the two edges of a cell face form a very sharp angle, these two edges may both pass through the particle. For this situation, a similar procedure as described above can be applied for each edge yielding the fractional volumes.

Case 4 : a node or vertex V that is common to the cell C_0 and its neighboring cells C_n, is occupied by a particle. The calculation of the volume V_n of the particle shared by C_n is performed as follows. First the vertex V is identified by checking all the vertices of the cell C_0. The intersection points of each edge (sharing V) with the particle surface are calculated. For cell C_n, these intersection points are denoted by P, Q and R, since there are only three edges between the three faces (denoted by f_1, f_2, f_3) for a tetrahedral, wedged

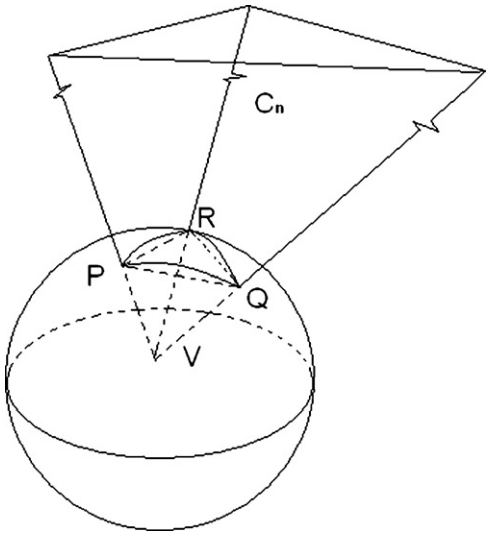

Fig. 4. Calculation of porosity on 3D unstructured mesh (case 4).

or hexahedral cell that are intersected by the particle (see Fig. 4). Then the particle volume V_k encompassed by the face PQR and f_k ($k = 1, 2, 3$) can be calculated by the approach described above in case 3 for the sphere volume surrounded by any two faces. Finally we obtain:

$$V_n = V_t + \left(V_{sg} - \sum_{k=1}^{3} V_k \right) \tag{26}$$

$$V_t = \frac{1}{6} |(PR \times PQ) \times PV|$$

where V_{sg} is the volume of the sphere segment split by the face PQR.

The analytical method presented above, though accurate, is relatively time-consuming compared with the point approximate method. This is due to the many arcsine and arctangent expressions that have to be evaluated for case 3 and case 4. To avoid evaluating repeatedly these complex expressions when calculating the void fraction at each time step, we introduced the look-up table. It consists of a pre-computed mapping table to accelerate the computation of the analytical solution. More about the look-up table method and the development of the analytical method for 2D unstructured mesh can be found in Wu et al. [26].

4.2. Implicit two-phase coupling

The coupling between the gas and particle phases is accomplished through the inter-phase volume fraction exchange and the inter-phase momentum transfer. Since the particle motion is resolved in the Lagrange framework, the volume fraction exchange due to particle motion appears only in the gas continuity and momentum equations. The resulting source term can be calculated very precisely, since the void fraction is accurately computed according to the method described in Section 4.1. The discretized source term in the gas mass conversation equation is given as follows:

$$S_c = -\frac{\rho_g}{\varepsilon^{n+1}} \left(\frac{\varepsilon^{n+1} - \varepsilon^n}{\Delta t} + \mathbf{u}_g^{n+1} \cdot \nabla \varepsilon^{n+1} \right) \tag{27}$$

The gas momentum source due to volume fraction exchange (the first part in the right hand side of Eq. (20)) is also calculated accurately.

The semi-implicit scheme is used to discretize the particle momentum equation (Eq. (4)). The nonlinear drag coefficient is evaluated explicitly, which results in a system of linear equations. When only buoyancy and two-phase drag forces are taken into account, the particle velocity is calculated as follows:

$$m_p \frac{\mathbf{u}_p^{n+1} - \mathbf{u}_p^n}{\Delta t} = -V_p (\nabla p)_p^{n+1} + \frac{V_p \beta_p^n}{1 - \varepsilon_p^{n+1}} (\mathbf{u}_{g,p}^{n+1} - \mathbf{u}_p^{n+1}) + m_p \mathbf{g} \tag{28}$$

where the Eulerian properties such as void fraction ε, static pressure gradient $\triangledown p$ and gas-phase velocity $\mathbf{u}_g$ are mapped to the particle position through gradient interpolations. Because the collocated finite volume discretization of the transport equations is employed in FLUENT, all Eulerian scalar values as well as their gradients are defined at the cell centre. The scalar gradients are computed using the divergence theorem:

$$\nabla \phi_r = \frac{1}{\Delta V} \sum_{f}^{N_f} \tilde{\phi}_f \mathbf{A}_f \tag{29}$$

where ΔV is the cell volume, and $\mathbf{A}_f$ is the normal vector pointing out of the cell face f. The face values $\tilde{\phi}_f$ are calculated by averaging ϕ from the two cells adjacent to face f. Then the scalar value at the particle position is expressed by

$$\phi_p = \phi_c + \nabla \phi_r \cdot \mathbf{dr}, \tag{30}$$

$$\mathbf{dr} = \mathbf{x}_p - \mathbf{x}_c$$

where $\mathbf{x}_c$ and $\mathbf{x}_p$ are the cell centroid vector and the particle position vector, respectively. To map the pressure gradient force to the particle position, the gradient for each component of the pressure gradient force is evaluated too. The above mapping method is very simple and more efficient and flexible, when compared to the bilinear or tri-linear interpolations [2,23]. The latter interpolation methods are only suitable for structured regular grids.

For the momentum source term due to forces exerted by the dispersed phase on the gas phase, the integration of Eq. (3) is taken over the grid cell:

$$\mathbf{S}_g = -\frac{1}{\Delta V} \sum_{k=1}^{NPC} \frac{V_{p,k} \beta_{p,k}^n}{1 - \varepsilon^{n+1}} (\mathbf{u}_g^{n+1} - \mathbf{u}_{p,k}^{n+1}) \tag{31}$$

where NPC is the number of particles associated with the cell, and $V_{p,k}$ are the corresponding fractional volumes. Consistent with the interpolation from the Eulerian grid to the particle centre, this formula implies that a particle may be acted on and also react to the fluid in several Eulerian cells. The source terms are linearized as said in Section 3.2 and the final linear coefficient is

$$B = -\frac{\rho_g}{\varepsilon^{n+1}} \left(\frac{\varepsilon^{n+1} - \varepsilon^n}{\Delta t} + \mathbf{u}_g^{n+1} \cdot \nabla \varepsilon^{n+1} \right) - \frac{1}{\Delta V} \sum_{k=1}^{NPC} \frac{V_{p,k} \beta_{p,k}^n}{(1 - \varepsilon^{n+1}) \varepsilon^{n+1}} \tag{32}$$

It can be seen from Eqs. (28) and (31) that all the particle momentum increase gained from the gas phase is feed-backed to the gas. That means a conservation of the momentum exchange due to the two-phase drag interactions. This owes to the combined facts: (i) the nonlinear drag force is linearized and the drag coefficient is evaluated explicitly; (ii) both the particle and gas velocities are calculated in the same time layer. Before iteration in a time step, the drag coefficient for each particle is first calculated according to the two-phase velocities in the last time step along with the gradient of the void fraction. Then, in each iteration for solving the gas pressure–velocity equations, the gradients of the involved Eulerian variables are calculated according to Eq. (29), and the linearized particle momentum equations are solved. At the same time, the

gas source terms due to the volume fraction exchange and interphase momentum transfer are evaluated in each iteration step. The iteration stops only when the solutions of both the gas mass and momentum equations converge.

It must be noted that the gas momentum source is calculated implicitly. The linear coefficient including the part due to two-phase drag interactions, in the source term, is automatically absorbed into the principle diagonal elements of the AMG solver matrix, which physically enhances the two-phase coupling and numerically helps the solution convergence. The particle drag force exerted on the gas is weighted by the particle fractional volume, which has a smooth effect on the source forces and helps numerical stability.

The solution of the particle velocity and the gas field is synchronized in FLUENT, resulting in a strongly implicit two-phase coupling. The present coupling algorithm differs significantly from that presented by Hoomans et al. [2]. In the latter, the two-phase momentum equations are solved in a segregated way, which means an explicit two-phase coupling. Moreover, in Ref. [2], the particle velocities are mapped back to the Eulerian grid and the gas source term is evaluated according to Eulerian properties, which is not quite in-line with the discrete feature of the particulate phase.

4.3. Data structure for collision events

Detection and handling of potential collision events in hard-sphere DPM often cause a heavy computational burden. Relatively little progress has been achieved in the development of the hard-sphere DPMs regarding the handling of the collision events compared to other aspects such as collision dynamics and fluid–particle interactions. Hoomans [27] discussed several optimization strategies for collision event handling in the context of DPM. A reduction in computation time was achieved using these strategies yet the complexity is of order $O(N_p)$ only because no proper data structure was introduced to handle the event list. Likewise, Hoomans [27] presented a simple algorithm without a complex data structure that resulted in $O(N_g)$ complexity. N_g is the number of cells in the domain. To the best knowledge of the authors, the $O(\log N_p)$ priority queue algorithms that are widely used in MD simulation [28–30], have not been implemented in the DPM framework to determine the current collision events. The aim of this sub-section is to present a new efficient approach to build a complexity $O(1)$ priority queue suitable for hard-sphere discrete particle modeling. The chained hash-table concept is introduced to speedup both insertion of collision events and location of the current event with the minimum collision time. The crosswise linking data structure for event list is used to determine and delete all invalid events associated with any colliding particle pair.

The basic algorithm for collision event handling is described for a system of N_P particles denoted by $p = \{0,1, \ldots, N_p - 1\}$. The system P occupies a simulation domain that is discretized into N_g finite volume grids. A collision event involving two particles m and n is denoted by $C_{m,n} = (m, n, t_{m,n})$ where $t_{m,n}$ is the collision time for the particle pair (m, n). The collision time can be calculated according to the positions of the particle pair [2,24]. A variable t_c is introduced to mark the time point at which the current collision is handled. The algorithm to handle all possible collisions in a given time step Δt is summarized as follows:

S1. Scan all particles and build up the collision event list, set $t_c = 0$;
S2. If there is no event in the event list, go to the next simulation time step; else
S3. Determine the current event $C_{m,n}$ having the minimum collision time in the list;
S4. $t_c = t_c + t_{m,n}$, update particle positions: $\mathbf{x}_p = \mathbf{x}_p + \mathbf{v}_p t_{m,n}$
S5. Update velocities of the current pair (m, n) using collision dynamics;
S6. Detect future possible events for the current pair (m, n);
S7. Update the event list. Go to S2.

If N_c is the number of collisions handled within a time step, it is evident that S2–S7 will be repeated N_c times in that time step. For the same flow conditions, this frequency should be proportional to the particle number N_p, so at this stage we just assume that N_c is of the same order as N_p. It means that if millions of particles are involved in the computation, steps S2–S7 will be computed millions of times. Thus one should pay close attention to these steps while optimizing the algorithm. Indeed, any loop in these steps must be avoided as far as possible. Though time-consuming, all particles must be scanned when initializing the collision list in S1. We introduce the complexity in time to illustrate a rough evaluation of the looping times for a given operation. In the following discussions, the computation complexity of S1 is evaluated for one particle, while those of S2–S7 are evaluated for the handling of one collision event, that is, for one collision iteration.

By introducing a neighbor list for each particle, only the neighborhood particles are scanned to detect potential collisions. Thus the complexity for S1 and S6 is $O(N_{nb})$. N_{nb} is the average particle number in the neighbor list. A neighbor list is constructed as follows:

$$Partner_m = \{n | \forall n \in P, |\mathbf{x}_n - \mathbf{x}_m| < R_s, m \neq n\}$$

It can be compiled for each particle $m \in P$ before initializing the collision list. For a mono-disperse system, R_s can be determined by

$$R_s = d_p + 2|\mathbf{v}|_{\max} \Delta t$$

$|\mathbf{v}|_{\max}$ is the maximum particle velocity.

Since there is no loop involved in S2 and S5, the complexity for these two steps is $O(1)$. By introducing a time stamp for each particle to mark the last updating time point, only the current colliding pair needs updating of their positions. So the complexity of S4 is also $O(1)$. When updating the event lists in S7, all events related to the current pair and their neighboring particles should be updated. If the events in all lists are arranged in an unsorted way, the complexity order of S7 should be $O(N_{nb})$. Applying the local minimum algorithm in MD studies [28,29], the individual event list associated with the particle, is constructed as follows:

$$LP_m = \{(m, n, t_{m,n}) | \forall n \in Partner_m\}$$

A global list having N_p elements is composed of the event with the smallest collision time in LP:

$$GLP = \{(p, n, t_{p,n}) | \forall p \in P, (p, n, t_{p,n}) \in LP_p \text{ and } \forall (p, m, t_{p,m}) \in LP_p, t_{p,n} < t_{p,m}, m \neq n\}$$

If a complete binary tree (CBT) data structure is employed for GLP, the $O(1)$ and $O(\log N_p)$ complexity can be achieved for S3 and S7, respectively. Otherwise, only the $O(N_p)$ complexity can be achieved for either S3 or S7, if the events in GLP are stored in a linear linked list.

Hoomans [27] utilized the finite difference grid and introduced a global list for all cells together with local lists for each cell. The complexity of operations on global list in S3 is close to $O(N_g)$. Usually, the number of grid cells N_g is at least one order less than that of the particles N_p. This strategy results in a substantial cut in the CPU time dedicated for handling collision events in DPMs, compared to the $O(N_p)$ algorithm. However, in this strategy, all cells including those that do not contain any particle and/or those where no collision is taking place are scanned. These cases are often encountered in heterogeneous fluidized beds. In addition, finding the event

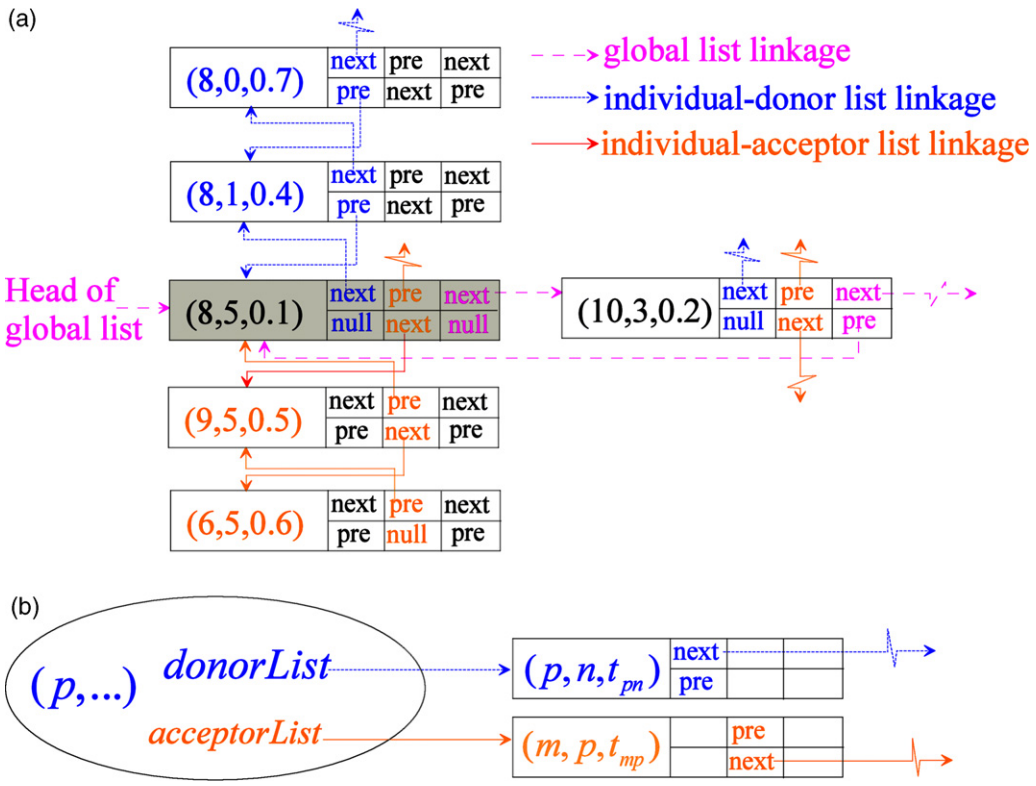

Fig. 5. Illustration of the crosswise linking of collision events: (a) three kinds of linkage are maintained for a given collision event (8, 5, 0.1). "Next" and "pre" are pointers to the next and preceding objects, respectively. "null" means a zero pointer without any object pointed to it; (b) Two pointers for a given particle p are responsible for maintaining the events in the donor and acceptor lists of particle *p*.

with the smallest collision time in the cell hosting the collision pair brings additional computational overhead. Since a particle is also a partner of its partner, it's evident that the collision partners as well as the collision time are stored redundantly. The duplicate storage of collision events also discounts the efficiency of the algorithms mentioned above.

To avoid additional overhead brought up by the redundant storage of the potential collision events, we first introduce a crosswise linking data structure for events in collision list. Three linkages for a given event $C_{p,q} = (p,q,t_{p,q})$ are built up and maintained when updating relevant collision lists. The C-Language pointer is introduced to access the linked event. The pointer variables for these linkages are all defined in the event data structure, which means that the event object is self-explained or any information related to the collision can be accessed from the pointer to the event object. This allows the unique storage for each event in the lists said above. As shown in Fig. 5, the first linkage is the individual-donor list for particle p, LPD_p, which is defined as the set of potential collision events detected when particle p is the host. The second one is the individual-acceptor list for particle q, LPA_q, which is defined as the set of collision events detected when the partner of particle q is the host. The third one is used for the global list. Thus, a given event $C_{p,q} = (p,q,t_{p,q})$ is certainly contained within the lists LPD_p and LPA_q, and may be contained in the global list *GLP*. The collision event is crosswise linked into these lists through a doubly linking structure (see Fig. 5a). Thanks to this type of structure, the previous and the next element can be determined without scanning from the list head during the element deleting and updating operations. Because a given particle p can be a host or an acceptor, it is always associated with both the donor list LPD_p and the acceptor list LPA_p. Two pointer variables associated with the particle p are allocated to store the memory locations of the head elements of these two lists, respectively, as shown in Fig. 5b. When operating collision events (inserting into or deleting from a list), we always take the first particle in the pair as the host and the second as the acceptor. This makes the updating operation of the donor and acceptor lists straightforward. For instance, when initializing the individual lists in S1, we just need to scan the particle p with its partners whose indexes are smaller than p, to build the list LPD_p:

$$LPD_p = \{(p, n, t_{p,n}) | \forall n \in Partner_p, p > n\}$$

At this stage, the acceptor lists LPA_n of its partners are built up too. For particle p's partners that their indexes are larger than p, we get

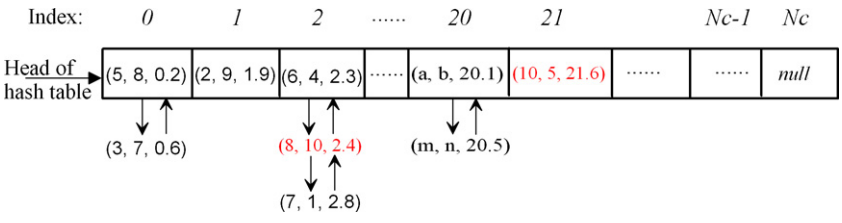

Fig. 6. Logical view of a chained hash table. When two or more events are mapped to the same location in the hash table, they are chained in a doubly linked list. The collision time is normalized by Δtc.

the list LPA_p:

$$LPA_p = \{(n, p, t_{p,n}) | \forall n \in Partner_p, p < n\}$$

By taking advantage of the cross-linking structure of the collision events, any relevant particle and/or collision event can be easily identified without searching in the corresponding individual lists. Since there is only one entity of any event, there is no additional scanning to update the individual lists for the partners of the colliding pair. Moreover, by sorting the elements in the individual lists, only the first element (local minimum one) in the donor list is added to the global one so that there is no duplicate event in the global list. The additional operation on the global list is avoided too. Thus, in S7, before building up the individual lists for the colliding pair, their corresponding events in the old lists are invalidated and cleaned, resulting in an efficient update of the local event lists and an $O(1)$ complexity of this step.

In computer science, a hash table, or a hash map, is a special data structure that associates keys with records. The primary operation it supports efficiently is a look-up: given a key, find the corresponding record. It works by transforming the key using a hash function into a hash code, a number that is used as an index in an array to locate the desired location where the record should be. Hash tables support the efficient insertion of new entries, in expected complexity of $O(1)$ in time.

Herein a chained hash table is introduced for the storage of the event in the global list *GLP* as shown in Fig. 6. The one-dimensional array is used to implement the hash table. The key of the hash table is the collision time $t_{m,n}$ and the record is the memory location pointing to the corresponding event in the individual-donor list. We implement the individual-donor list LPD as a priority queue so that only its first element is mapped/inserted into the hash table. The individual-acceptor list is implemented as an unsorted linked list since there is no need to pick events from this list (an $O(1)$ complexity is achieved for insertion and deletion on this list).

To satisfy the priority queue requirement of having the event with the smallest collision time accessed first in the hash table, the hash function should be well designed to map orderly the collision times to integers or indices I and to locate the position where the events are in the table. A simple hash function is defined as

$$I(t_{m,n}) = INT(t_{m,n}/\Delta t_c) \text{ with } \Delta t_c = \Delta t/N_c, \quad (33)$$

where the function $INT(x)$ gives the integral part (lower integer) of x. If the particles collide successively in a uniform time interval, the above hash function maps only one element to a table location. When inserting a collision event, its address or index in the global list is determined by the hash function without any loop and the expected computational complexity in time will be reduced to $O(1)$.

However, one cannot determine in *a priori* manner the number of collisions in a time step. Considering the slow change of a given gas–particle system through the small time steps that are often used in DPM simulations (at least one order less than the particle response time), the number of collision occurrences for successive time steps should change very little. As a direct consequence, the length/capacity of hash table will also vary little. In fact, the number of collisions N_c in Eq. (33) is actually selected as that in the last time step N_c^0. Another fact is that the particles in gas–solid system may never collide in a uniform time interval, which may generate a same index using the hash function Eq. (33). This conflict problem can be solved by chaining [31]. By the chaining methodology, the hash table looks like a two-dimensional array with a variable length for the second dimension, or like a one-dimensional array of priority queues, as shown in Fig. 6. Because the length of the hash table is chosen properly and it is a nearly continuous function of the collision number in gas–solid system, chaining events is very rare process (in real simulations, the number of events in the chained lists is often less than 3). Thus the chained hash-table strategy gives exactly an $O(1)$ complexity in computational time for event insertion.

As discussed above, there are three types of linkage to be maintained for any given event. The third linkage for the global list *GLP* is used here for chaining in the hash table, as shown in Fig. 6. The event address/hash code in the hash table is stored with the corresponding event using an integer variable. When the head event $C_{a,b}$ in the donor list LPD_a is inserted to the hash table, its linkages in both the donor list LPD_a and the acceptor list of the particle a are kept. That means for any given event, whenever it is mapped to the hash table, its three linkages are retained. If $C_{a,b}$ becomes the current event, it will be released to the event pool while handling it, together with all other elements in LPD_a. Otherwise if there is an event $C_{b,m}$ occurring before $C_{a,b}$, $C_{a,b}$ will be released together with all other elements in LPA_b because particle b is the acceptor part of the collision when the event $C_{b,m}$ is processed. In the former case, there is no additional operation on the hash table since only the head event $C_{a,b}$ in LPD_a is mapped to the hash table. In the latter case, a null pointer is directly assigned to the location in the hash table indicated by its hash code if any element in LPA_b (such as $C_{a,b}$) has been mapped to the hash table. As shown in Fig. 6, the events (8, 10, 2.4) and (10, 5, 21.6) are marked invalid since they are associated to the individual lists of the current colliding pair (5, 8). Event (8, 10, 2.4) can be released from the chained hash table by re-linking its previous and next objects maintained in its third linkage pointers. Event (10, 5, 21.6) can be released directly by assigning a null pointer in the location (21) of the hash table. By the crosswise linking data structure, all events in the individual lists can be deleted without any searching. Thus the complexity for event deletion is also $O(1)$. Because there is no duplicate event, the deleting operation is minimized. The simulation is forwarded by scanning the entire hash table from the beginning. This can be done in a simple loop. The loop indicator presents the location of the current event in the hash table. When the last recode of the hash table is null, it means there is no longer event in this time step. Since the capacity of the hash table is chosen close to N_c, the expected complexity to

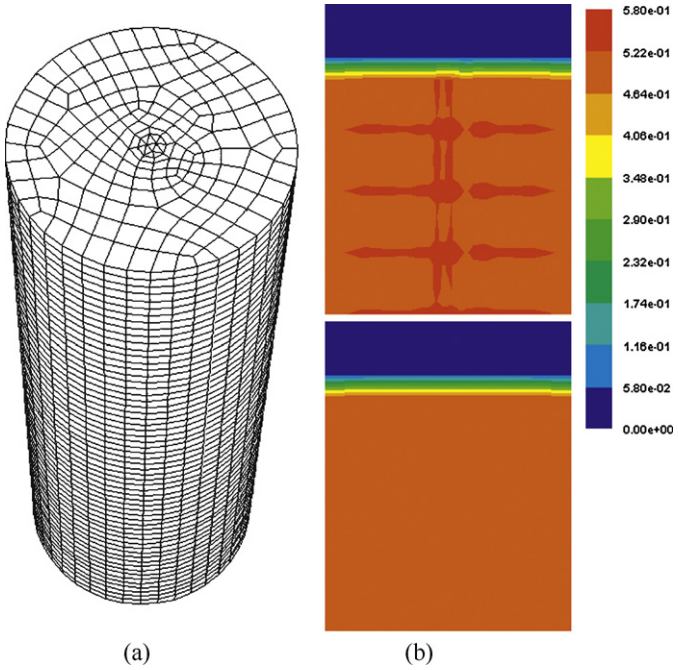

Fig. 7. (a) Grid mesh of the cylindrical bed; (b) the particle volume fraction at the centre of the cross-section calculated by simple method (top) and accurate method (bottom).

Table 1
Diagnostic numerical simulation parameters.

Bed diameter	0.2 m
Bed height	0.6 m
Stagnant bed height	0.2 m
Centre jet diameter	0.02 m
Minimum fluidization velocity U_{mf}	0.85 m/s
Background inlet gas velocity	$1.2U_{mf}$
Centre jet gas velocity	$20U_{mf}$
Particle diameter	1.5 mm
Particle density	2400 kg/m^3
Cell number	11,580
Particle number	1,770,000
Restitution coefficients e_n, e_t	0.95, 0.4
Friction coefficient μ_f	0.2
Time step Δt	0.0001 s

Table 2
CFB simulation parameters.

Riser diameter	0.1 m
Riser height	1.0 m
Superficial gas velocity	5.46 m/s
Particle diameter	800 μm
Particle density	2520 kg/m^3
Grid number	17,440
Particle number	521,000
Restitution coefficients e_n, e_t	0.97, 0.35
Friction coefficient μ_f	0.1
Time step Δt	0.0001 s

locate the current event is also $O(1)$. In summary, by resorting to the chained hash table methodology, all operations of collision events are performed at the computation complexity of $O(1)$.

5. Performance and capability of the model

Since many researchers [2,10,23,24] have verified and validated the hard-sphere DPM by comparing numerical results with experimental observations and features of the model have been revealed, herein attention is paid to the new characteristics of the model from the viewpoints of performance, capability and applicability, particularly when applied in conjunction with unstructured meshes. In the following test cases, the inter-phase drag force is calculated by the Gidaspow's drag formula [17]. All the numerical simulations were carried out on a PC platform (Intel Corel E6600^{TM} series).

5.1. Single bubble in cylindrical bed

To examine the model efficiency, several diagnostic simulations of a cylindrical bed were conducted. The cylindrical domain is meshed by unstructured hexahedral cells as shown in Fig. 7a. In these numerical experiments, the particle number is large, reaching about 1,800,000. The numerical parameters are specified in Table 1. At the beginning, the particles are distributed uniformly at the bed bottom with a porosity of about 0.52. Although the average cell size is about 300 times bigger than that of a particle, the distribution of the void fraction in the stagnant bed (at the vertical cross-sectional plane $y = 0$) calculated by the proposed method is much more uniform than by a simple way of neglecting the split of the particle volume between adjacent cells, as shown in Fig. 7b. Besides, the number of iterations required for the solution of the two-phase momentum equations to converge is reduced from an average of 24 ± 2 to 16 ± 2; it was reduced by about 33% in each time step, which saves 15–25% of the CPU time. When the look-up table of the void fraction was used, additional 2–5% CPU time on calculation of the two-phase momentum equations was saved. The void fraction is very important to the gas mass balance, so a smooth and accurate distribution ensures numerical stability and speeds up the solution convergence. It is also a key parameter to estimate the gas–solid drag forces. So from both the physical and numerical points of view, the void fraction should be calculated accurately in discrete particle simulations.

Fig. 8 shows the computational cost linked to the particle collision dynamics. The result is presented as a ratio of the computational time consumed in a simulation that uses the chained hash table strategy to the one necessary for a simulation without the said strategy. The data of the first 160 simulation time steps is selected for comparison. When a large number of collisions is involved in a time step (especially when it is equal to or larger than the particle number), it can be seen, that about 90% of the CPU time is saved if the hash table is used. In these simulations, the maximum CPU time consumed in a time step is about 256 s if the hash table strategy is not applied, but is reduced to less than 20 s when it is used. Similar 2D simulations were performed and we found out that up to 98% of the CPU time is saved if the hash table storage for the

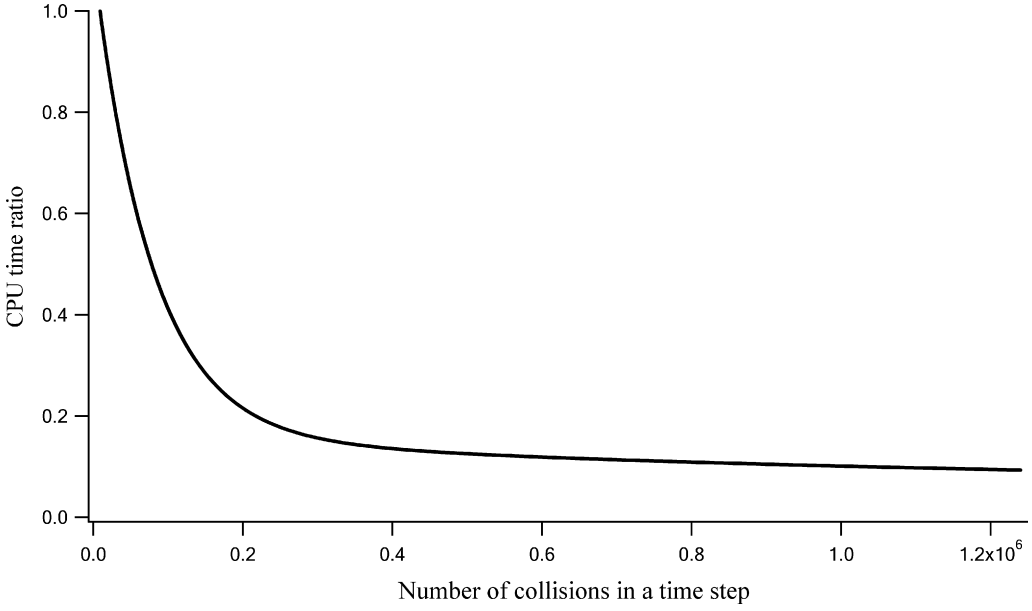

Fig. 8. Ratio of the CPU time for computing particle collisions using chained collision events table to that using linear global list.

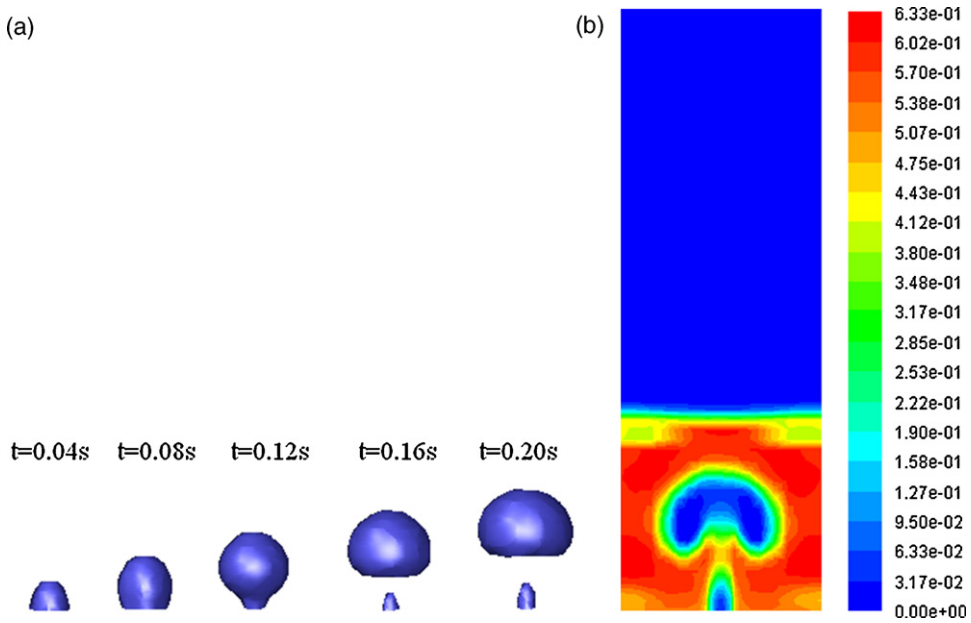

Fig. 9. 3D bubble shape simulated by DPM: (a) bubble formation (the iso-surface represents $\varepsilon = 0.8$) and (b) particle volume fraction distribution at the centre of the cross-section plane at $t = 0.2$ s.

collision list is applied. This is because the 2D collision dynamics is simpler than that of the 3D, resulting in a higher computational weight regarding the handling of event lists. Based on our numerical experiments, the time to simulate a 3D dense particulate flow with more than 500,000 particles is not acceptable if the hash table strategy is not used. It should be noted that the collision dynamics implemented in this study is a rigorous hard-sphere model without particle overlap, which is different from that by Helland et al. [14]. The 3D model can correctly predict the bubble shape change from elliptic to spherical as the bubble wake gradually forms as shown in Fig. 9, which agrees with the experimental observations [32].

5.2. Core–annulus flow structure in cylindrical riser

The gas–solid flow hydrodynamics in CFB risers has attracted a lot of research interest in recent decades because of its wide applications in industry. The granular flow in a CFB riser is often featured by a high concentration of solids flowing downward near the walls and a core dilute region with upward flowing solids. This is the so-called core–annulus flow structure that has been reported by many experimental studies [33–36]. Although the Eulerian two-fluid model is often used in numerical studies, the discrete particle model has the potential to clarify many of the flow characteristics in the CFB risers [11,13–15]. However, 3D discrete particle simulations of the cylindrical riser have not been successfully implemented, so far to the best of the authors' knowledge. The 2D DPM under-estimates the particle–wall interaction in quasi-2D problems and hence may not be able to simulate accurately the heterogeneous structure in real 3D CFB risers.

A laboratory scale riser with a diameter of 0.1 m and a height of 1 m is simulated by the present 3D model. Particle properties and numerical parameters are given in Table 2. It is relatively difficult to

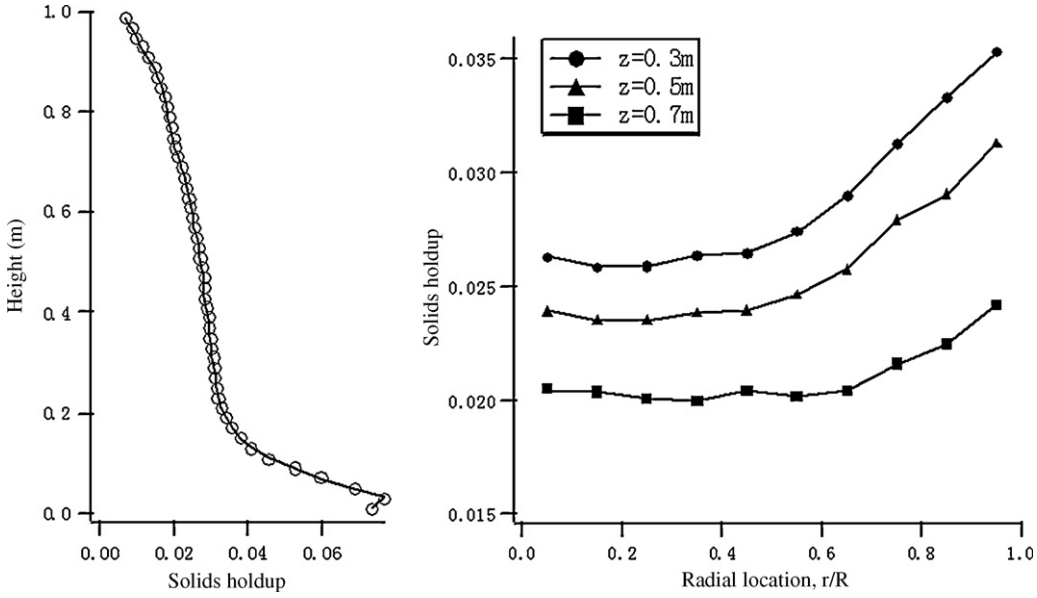

Fig. 10. Time-averaged axial distribution and radial profiles of particle holdup.

analyze the numerical diffusion for DPM than for TFM, since the former is related to the two-phase coupling algorithm. For two-phase flow simulated by TFM or general single-phase flow simulated by a finite volume method, the numerical diffusion is mainly caused by the mesh quality and the numerical scheme used to calculate the scalar value at the cell faces when integrating the transport equations. While for DPM, the source terms in the momentum equations of the continuous phase may cause additional diffusions, because the momentum exchange is evaluated not only by the cell scalar values, but also related to the particle velocities in the Lagrangian coordinates. It is well known that the use of hexahedral cells results in less numerical diffusion than the tetrahedral cells since more faces are involved to evaluate the scalar gradient. Thus, to reduce the numerical diffusion, the unstructured hexahedral grid is adopted to mesh the domain and the second-order upwind scheme is used to calculate the scalar value at the cell face. Gas-phase turbulence is not taken into account. At the beginning, the particles are distributed uniformly inside the cylindrical column with an average voidage of 0.982. The particles are introduced to the inlet gradually over several time steps to void numerical instability after travelling out of the riser outlet. The simulation is run for 12 s real time. To reduce the influence of the initial conditions, time-averaged results are obtained for the last 6 s of the simulation. The CPU time is about 150 h for one simulation run.

Fig. 10 shows the time-averaged axial solids holdup and the radial profiles of the particle volume fraction at different bed levels. It can be seen that the dense regime exists at the riser bottom below 0.1 m, with the solids fraction between 0.05 and 0.08, about two to four times the average solids holdup. Above $z = 0.2$ m, the fast fluidization regime can be noticed with an almost constant axial solids holdup. However, the radial particle distribution is not uniform and is characterized by the core–annulus flow structure with dense solids flowing downward nears the wall and a more dilute core flowing upward in the riser centre. The particle velocity distribution at two vertical cross-sections through the riser centre is shown in Fig. 11. It can be seen that the flow is not axi-symmetric and the particle vertical velocity is higher in the core region than in the annulus region. The particle radial profiles at different levels indicate that the higher the cross-sectional average solids holdup, the more non-uniform the radial distribution is, resulting in a thicker annular down-flow layer, which agrees well with the experimental observations [33,34].

The numerical model also captures the transient state of the heterogeneous structure such as particle clusters or sheets near the riser wall, as shown in Fig. 12. The snapshots reveal that the particle clusters or swarms form first near the wall. Two regions of particle circulation exist below and above the swarm (see Fig. 12b), which lead to a converged flow regime at the interface. The shear movement of particles in the interface elongates the swarm and forms the particle sheet. Because of the particle–wall interactions the particle sheet twists to form a U-shape. The particle sheet disappears and splits into discrete particles as the particle shear movement continues. The cluster shape and size are comparable to the observations reported by Horio and Kuroki [37]. Since the particle flow behavior can be resolved transiently at the individual particle level, the 3D discrete particle model presented here is a powerful tool for analyzing the heterogeneous structures taking place in CFB risers.

5.3. Spout–annulus flow structure in flat spouted bed

The gas–solid spouted beds can provide good mixing and contact between gas and coarse particles. They are often characterized by a dense region of downward particle flow called the annulus, a dilute core region of upward particle flow called the spout and the fountain region with centre upward and outer downward particle motion. Because the particle regime in the annulus is very dense and the particle–particle interactions are mainly contact-dominated, the discrete or distinct element model, in which the inter-particle interaction is described by the soft-sphere model, is popular in Lagrangian simulations of the spouted bed [38,39]. In these studies, the structured grid is adopted and the conical part of the bed is meshed in a stair-step way.

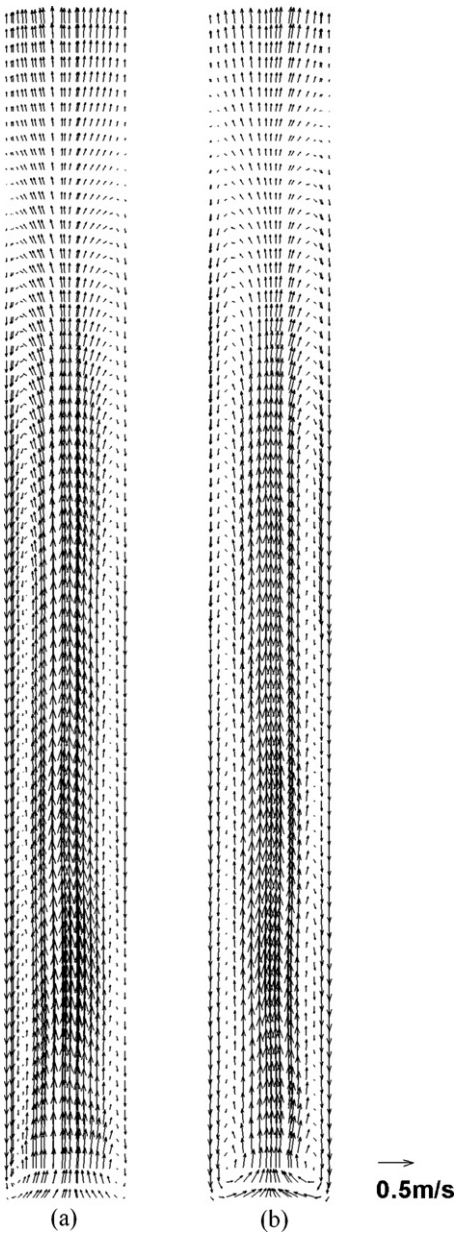

Fig. 11. Time-average particle velocity distribution at two cross-sections of the cylindrical riser: (a) $x = 0$ and (b) $y = 0$.

Zhao et al. [38] studied experimentally both the time-averaged and instantaneous flow behaviors of granular solids in a quasi-2D spouted bed and compared their 2D DEM simulation results with the experiment. We performed a 3D discrete particle simulation of the quasi-2D spouted bed using the proposed model. The bed dimensions and simulation parameters are the same as those of the experiment in Ref. [38] and the collision coefficients are selected as $\mu_f = 0.15$, $e_n = 0.95$, and $e_t = 0.35$. A hybrid mesh combining hexahedral and wedged cells is used with the total cell number of 6231 and the time step is set to 0.0001 s. Because there are steep gradients of the gas velocity near the interface between the spout and the annulus, the Saffman lift force due to gas shear flow is taken into

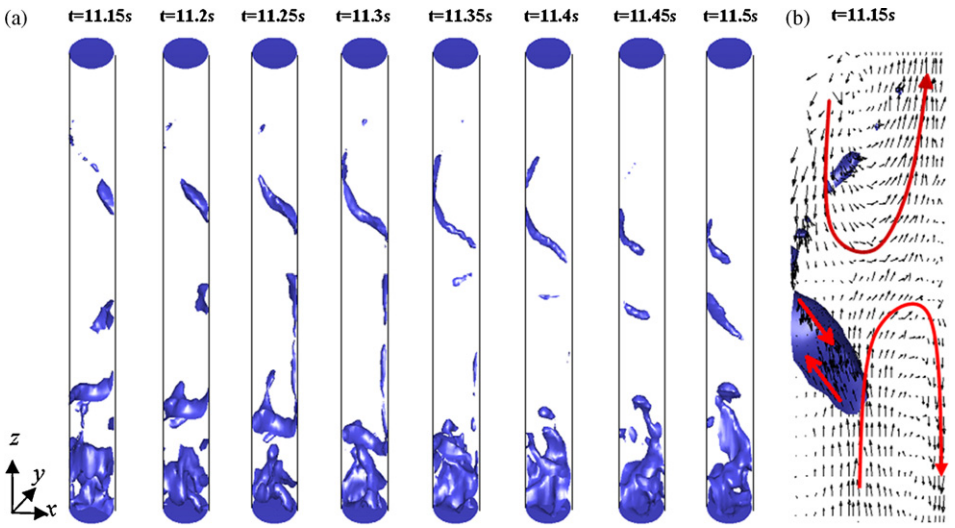

Fig. 12. Snapshots of heterogeneous structures in the riser simulated using 3D DPM: (a) particle cluster or swarm elongated to sheet with iso-surface representing dense phase and (b) cell-averaged particle velocity field at $x = 0$ cross-section between $z = 0.58$ m and $z = 0.88$ m.

account as follows:

$$\mathbf{F}_{Saff} = 1.61 d_p^2 (\mu \rho_g)^{1/2} |\boldsymbol{\omega}_g|^{-1/2} (\mathbf{u}_g - \mathbf{u}_p) \times \boldsymbol{\omega}_g$$

where $\boldsymbol{\omega}_g = \nabla \times \mathbf{u}_g$ is the gas vorticity.

Fig. 13 shows the flow pattern of particles at different points of time. The numerical simulation captures the main features of the spouted bed: the annulus, the spout and the fountain flow regions are distinguished very clearly. The transient granular flow is not steady but appears to be periodic. Also, a denser neck appears near the inlet, flows upward and finally disappears at the end of the spout. That means the particles do not move individually but as a cluster in the spouting region. While the particle cluster in the neck moves upward, it progressively grows bigger and denser with more particles entrained from the annulus. When it reaches the spout end, the spout is almost 'choked'. After that the cluster is injected into the fountain and the particles fall down to the annulus. The above flow behavior agrees well with the experimental observations [38,40]. The predicted period is about 0.11 s, which is slightly less than the experimental one (0.15 s). The time-averaged parti-

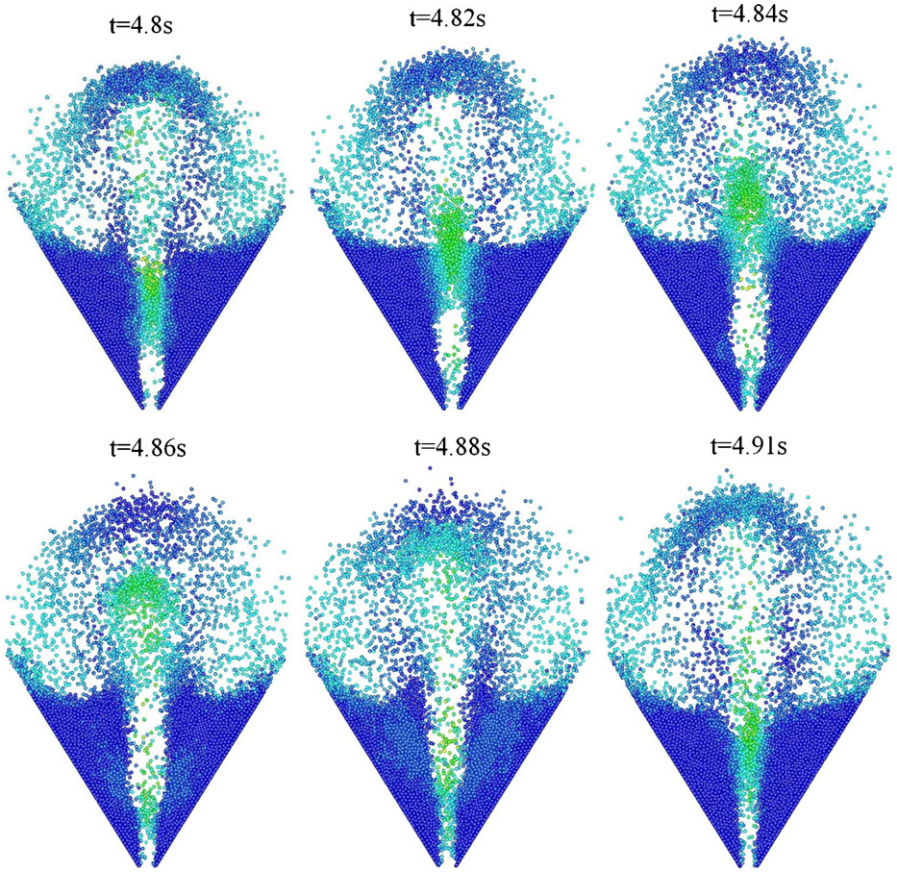

Fig. 13. Snapshots of particle distribution simulated by 3D DPM (color from blue to yellow denotes particle velocity magnitude from 0 to 1.1 m/s).

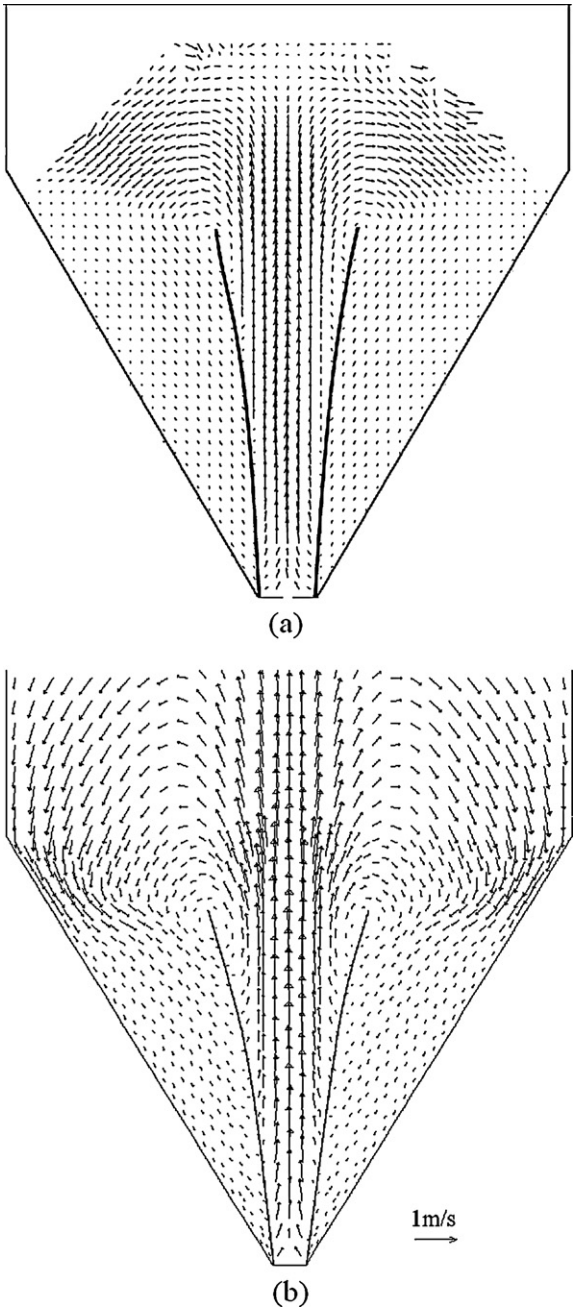

Fig. 14. Comparison of time-averaged particle velocity field: (a) experimental results by Liu et al. [40] (the top part of the fountain is not shown because of the small number of particles measured there, see Liu et al. [40]) and (b) 3D DPM results, region between thick lines (zero vertical velocity) is spout.

cle velocity simulated by the present 3D DPM is compared with that of the experiment, as shown in Fig. 14. Based on the particle velocity field, the interface separating the spout and the annulus can be delineated, where the particle time-averaged vertical velocity is zero. The predicted spout height is about 0.088 m, close to the experimental value of 0.08 m. The predicted time-averaged flow also agrees well with that of the experiment, but the particle velocity in the upper spout and fountain is over-predicted. This can also be noticed on the axial and lateral profiles of the particle vertical velocity as shown in Figs. 15 and 16. The particle vertical velocity reaches 1.1 m/s, larger than that of the experiment, at the spout end. The axial profile shows that the particles accelerate more in the lower spout than in the upper spout, which agrees with the

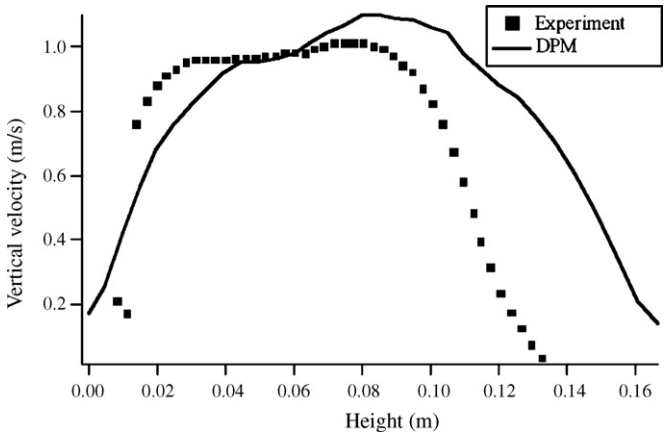

Fig. 15. Comparison of vertical particle velocity along the spout axis with that of experiment by Zhao et al. [38].

experimental observations. It can be seen that the particle vertical velocity reaches maximum in the spout centre and declines to zero which is defined as the interface separating the spout and the annulus. This is caused by two combined effects: (1) the centre gas jet has a maximum velocity which results in a maximum drag on the particle in the spout centre; (2) the flow is dilute in the spout centre and inter-particle frictional interaction (shear force) is less significant than in the interface. The expansion of the radial position of the vertical velocity reflects the expanding shape of the spout, which is also shown in Fig. 14. Particles accelerate almost in the whole spout region. This phenomenon is completely different from the traditional conical spouted bed, in which particles only accelerate for a very short period near the inlet orifice and gradually decelerate through the spout [41]. The difference may be caused by the formation of cluster in the spout. In conical spouted bed, no particle cluster or "chocking" was observed and the flow is dilute in the spout [41,42]. The voidage inside the forming cluster is considerably low which results in a higher drag force to the particles (the drag force is correlated to the voidage), which causes the particles in the cluster to accelerate. The lateral particle velocity profiles predicted by the 3D DPM show better agreement with those of the experiment than those by the 2D DEM [38]. The latter under-estimated the particle vertical velocity in the spout (see Fig. 5b in Ref. [38]), which was considered as a result of the simplification of the 2D simulation including errors in the calculations of porosity and/or drag force [38]. The overestimated spout height and particle velocity in the fountain may be caused by the Gidaspow's drag formula used in the present simulation, which over-predicts the two-phase

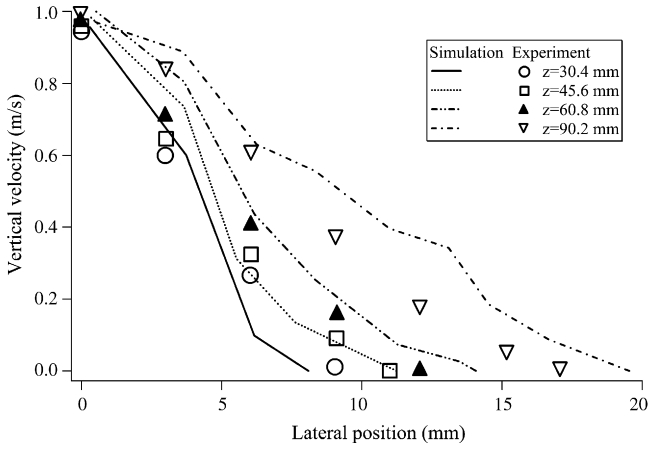

Fig. 16. Lateral profiles of vertical particle velocity at different levels.

drag force when the heterogeneous flow structure forms in the bed [13,19].

6. Concluding remarks

A robust and efficient three-dimensional discrete particle model based on unstructured grids was developed under the CFD platform FLUENT with the aim to simulate gas–solid systems with large number of particles. Compared with other discrete particle models in the literature, the model has the following advantages:

(1) For the general case of 3D unstructured meshes, the void fraction in the cell is calculated accurately when a particle is not contained wholly within one cell.
(2) The chained hash table strategy is applied together with the crosswise linking storage of the particle collision events, which greatly improves the computational performance of the collision model and allows the numerical simulation of gas–solid systems having a large number of particles.
(3) Strongly implicit two-phase coupling is adopted, which greatly enhances the numerical stability and speeds up the solution convergence.
(4) Special treatments are adopted to implement the present model under the basic incompressible flow solver of FLUENT and full customization of the model is achieved.

The numerical results show that the present model can capture many important features of the gas–solid flow systems, such as bubbles in a bubbling fluidized bed, the core–annulus structure and cluster in a CFB riser, and the spout–annulus structure in a spouted bed. It is important to mention that the present model cannot handle situations where the particle size is bigger than the grid size. An ongoing work is currently dedicated to enhance the capabilities of the algorithm to simulate any gas–particle system regardless the size of the grid or particles involved.

Acknowledgement

This work was partly supported by a grant from the Guangdong Natural Science Foundation.

References

[1] Y. Tsuji, T. Kawaguchi, T. Tanaka, Discrete particle simulation of two-dimensional fluidized bed, Powder Technology 77 (1) (1993) 79–87.
[2] B.P.B. Hoomans, J.A.M. Kuipers, W.J. Briels, W.P.M. van Swaaij, Discrete particle simulation of bubble and slug formation in a two-dimensional gas-fluidised bed: a hard-sphere approach, Chemical Engineering Science 51 (1) (1996) 99–118.
[3] N.G. Deen, M. Van Sint Annaland, M.A. Van der Hoef, J.A.M. Kuipers, Review of discrete particle modeling of fluidized beds, Chemical Engineering Science 62 (1–2) (2007) 28–44.
[4] Y. Tsuji, Multi-scale modeling of dense phase gas–particle flow, Chemical Engineering Science 62 (13) (2007) 3410–3418.
[5] M.A. Van der Hoef, M. van Sint Annaland, J.A.M. Kuipers, Computational fluid dynamics for dense gas–solid fluidized beds: a multi-scale modeling strategy, Chemical Engineering Science 59 (22–23) (2004) 5157–5165.
[6] M. Van Sint Annaland, N.G. Deen, J.A.M. Kuipers, Numerical simulation of gas–liquid–solid flows using a combined front tracking and discrete particle method, Chemical Engineering Science 60 (22) (2005) 6188–6198.
[7] E. Delnoij, F.A. Lammers, J.A.M. Kuipers, W.P.M. Swaajj, Dynamic simulation of dispersed gas–liquid two-phase flow using a discrete bubble model, Chemical Engineering Science 52 (9) (1997) 1429–1458.
[8] J. Ouyang, J. Li, Discrete simulations of heterogeneous structure and dynamic behavior in gas–solid fluidization, Chemical Engineering Science 54 (22) (1999) 5427–5440.
[9] M.J. Rhodes, X.S. Wang, M. Nguyen, P. Stewart, K. Liffman, Study of mixing in gas-fluidized beds using a DEM model, Chemical Engineering Science 56 (8) (2001) 2859–2866.
[10] G.A. Bokkers, M. van Sint Annaland, J.A.M. Kuipers, Mixing and segregation in a bidisperse gas–solid fluidized bed: a numerical and experimental study, Powder Technology 140 (3) (2004) 176–186.
[11] C.H. Ibsen, E. Helland, B.H. Hjertager, T. Solberg, L. Tadrist, R. Occelli, Comparison of multifluid and discrete particle modelling in numerical predictions of gas particle flow in circulating fluidised beds, Powder Technology 149 (1) (2004) 29–41.
[12] C.L. Wu, J.M. Zhan, Numerical prediction of particle mixing behavior in a bubbling fluidized bed, Journal of Hydrodynamics Series B 19 (3) (2007) 335–341.
[13] E. Helland, H. Bournot, R. Occelli, L. Tadrist, Drag reduction and cluster formation in a circulating fluidized bed, Chemical Engineering Science 62 (1–2) (2007) 148–158.
[14] E. Helland, R. Occelli, L. Tadrist, Numerical study of cluster formation in gas–particle flow in a circulating fluidized bed, Powder Technology 110 (3) (2000) 210–221.
[15] H.S. Zhou, G. Flamant, D. Gauthier, J. Lu, Lagrangian approach for simulating the gas–particle flow structure in a circulating fluidized bed riser, International Journal of Multiphase Flow 28 (2) (2002) 1801–1821.
[16] J. Li, J.A.M. Kuipers, On the origin of heterogeneous structure in dense gas–solid flows, Chemical Engineering Science 60 (5) (2005) 1251–1265.
[17] D. Gidaspow, Multiphase Flow and Fluidization–Continuum and Kinetic Theory Descriptions, Academic Press, San Diego, 1994.
[18] D.L. Koch, R.J. Hill, Inertial effects in suspension and porous-media flows, Annual Review of Fluid Mechanics 33 (2001) 619–642.
[19] W. Wang, J. Li, Simulation of gas–solid two-phase flow by a multi-scale CFD approach—extension of the EMMS model to the sub-grid level, Chemical Engineering Science 62 (1–2) (2007) 208–231.
[20] D.J. Mavriplis, Unstructured grid techniques, Annual Review of Fluid Mechanics 29 (1997) 473–514.
[21] Y. Zhao, H.H. Tan, B. Zhang, A high-resolution characteristics-based implicit dual time-stepping VOF method for free surface flow simulation on unstructured grids, Journal of Computational Physics 183 (1) (2002) 233–273.
[22] M. Jacob, ProCell technology: modelling and application, Powder Technology 189 (2) (2009) 332–342.
[23] M.J.V. Goldschmidt, R. Beetstra, J.A.M. Kuipers, Hydrodynamic modelling of dense gas-fluidised beds: comparison and validation of 3D discrete particle and continuum models, Powder Technology 142 (1) (2004) 23–47.
[24] C.L. Wu, J.M. Zhan, Y.S. Li, K.S. Lam, Dense particulate flow model on unstructured mesh, Chemical Engineering Science 61 (17) (2006) 5726–5741.
[25] Fluent Inc., FLUENT 6.3 User's Guide, ANSYS Incorporated, 2006.
[26] C.L. Wu, J.M. Zhan, Y.S. Li, K.S. Lam, A.S. Berrouk, Accurate void fraction calculation for three-dimensional discrete particle model on unstructured mesh, Chemical Engineering Science 64 (6) (2009) 1260–1266.
[27] B.P.B. Hoomans, Granular dynamics of gas–solid two-phase flows, PhD thesis, Twente University, Netherlands (2000).
[28] M. Marin, D. Risso, P. Cordero, Efficient algorithms for many-body hard particle molecular dynamics, Journal of Computational Physics 109 (1993) 306–317.
[29] M. Marin, P. Cordero, An empirical assessment of priority queues in event-driven molecular dynamics simulation, Computational Physics Communication 92 (1995) 214–224.
[30] H. Sigurgeirsson, A. Stuart, W.L. Wan, Algorithms for particle-field simulations with collisions, Journal of Computational Physics 172 (2001) 766–807.
[31] M.T. Goodrich, R. Tamassia, D.M. Mount, Data Structures and Algorithms in C++, Wiley & Sons Inc., 2004.
[32] R. Clift, J.R. Grace, Continuous bubbling and slugging, in: Davidson, Clift, Harrison (Eds.), Fluidization, 2nd edition, Academic Press, London, 1985.
[33] D. Bai, E. Shibuya, Y. Masuda, K. Nishio, N. Nakagawa, K. Kato, Distinction between upward and downward flows in circulating fluidized beds, Powder Technology 84 (1) (1995) 75–81.
[34] H.T. Bi, J.R. Grace, Flow regime diagrams for gas–solid fluidization and upward transport, International Journal of Multiphase Flow 21 (6) (1995) 1229–1236.
[35] M.J. Rhodes, M. Sollaart, X.S. Wang, Flow structure in a fast fluid bed, Powder Technology 99 (2) (1998) 194–200.
[36] S.W. Kim, G. Kirbas, H.T. Bi, C.J. Lim, J.R. Grace, Flow structure and thickness of annular downflow layer in a circulating fluidized bed riser, Powder Technology 142 (1) (2004) 48–58.
[37] M. Horio, H. Kuroki, Three-dimensional flow visualization of dilutely dispersed solids in bubbling and circulating fluidized beds, Chemical Engineering Science 49 (15) (1994) 2413–2421.
[38] X.L. Zhao, S.Q. Li, G.Q. Liu, Q. Yao, J.S. Marshall, DEM simulation of the particle dynamics in two-dimensional spouted beds, Powder Technology 184 (2) (2008) 205–213.
[39] S. Takeuchi, S. Wang, M. Rhodes, Discrete element method simulation of three-dimensional conical-base spouted beds, Powder Technology 184 (2) (2008) 141–150.
[40] G.-Q. Liu, S.-Q. Li, X.-L. Zhao, Q. Yao, Experimental studies of particle flow dynamics in a two-dimensional spouted bed, Chemical Engineering Science 63 (4) (2008) 1131–1141.
[41] M.J. San José, M. Olazar, S. Alvarez, M.A. Izquierdo, J. Bilbao, Solid cross-flow into the spout and particle trajectories in conical spouted beds, Chemical Engineering Science 53 (20) (1998) 3561–3570.
[42] M.J. San José, M. Olazar, S. Alvarez, M.A. Izquierdo, J. Bilbao, Local bed voidage in conical spouted beds, Industrial & Engineering Chemistry Research 37 (6) (1998) 2553–2558.