A parallel fictitious domain method for the interface-resolved simulation of particle-laden flows and its application to the turbulent channel flow

Zhaosheng Yu[1], Zhaowu Lin[1], Xueming Shao[1,*], Lian-Ping Wang[2]

[1] State Key Laboratory of Fluid Power Transmission and Control, Department of Mechanics, Zhejiang University, Hangzhou 310027, China

[2] Department of Mechanical Engineering, University of Delaware, Newark, Delaware 19716, USA

Abstract: A parallel direct-forcing fictitious domain method for the simulation of particulate flows is reported in this paper. The parallel computing strategies for the solution of flow fields and particularly the distributed Lagrange multiplier are presented, and the high efficiency of the parallel code is demonstrated. The new code is then applied to study the effects of particle density (or particle inertia) on the turbulent channel flow. The results show that the large-scale vortices are weakened more severely, and the flow friction drag increases first and then reduces, as particle inertia is increased.

Keywords: Fictitious domain method; Parallel; Particle-laden flows; Turbulent channel flow

1 Introduction

Particle-laden flows are ubiquitous in nature and industrial settings such as fluidized bed reactor and slurry transport. The two-fluid approach and Discrete Particle Model (DPM) based on the point-particle approximation are two traditional methods commonly used to treat the particle-laden flows in engineering applications, whereas the interface-resolved simulation method has been developed to advance mechanistic understanding in multiphase flows in the past two decades (Balachandar and Eaton, 2010; Tryggvason, 2010). In the interface-resolved simulation method, the no-slip boundary condition on the particle-fluid interface is considered and the hydrodynamic force on the particles is determined via the solution of the flow fields outside the particle boundaries. Unlike the two-fluid and discrete particle models, the interface-resolved method computes directly the hydrodynamic force on the particles, and therefore is also called a direct numerical simulation method for the particle-laden flows (Hu et al., 2001).

There exist different interface-resolved solvers such as the arbitrary Lagrangian-Eulerian method (Hu et al., 2001), lattice Boltzmann method (Ladd, 1994), fictitious domain method (Glowinski et al., 1999) and immersed boundary method (Uhlmann, 2005). The fictitious domain method for the particulate flows was proposed by Glowinski et al. (1999), and the key idea in this method is that the interior domains of the particles are filled with the same fluids as the surroundings and the distributed Lagrange multiplier (i.e. a pseudo body force) is introduced to enforce the interior (fictitious) fluids to satisfy the constraint of rigid-body motion (Glowinski et al., 1999). Yu and his coworkers have made new implementations of the fictitious domain method and conducted many successful applications with a serial code (Yu et al., 2004, 2006; Yu and Shao, 2007, 2010; Shao

[*] Corresponding author. E-mail address: mecsxm@zju.edu.cn (X. Shao)

et al., 2008, 2012). The primary aim of the present study is to develop an efficient parallel code of the fictitious domain method. We note that the parallelizations of the immersed boundary methods or the fictitious domain methods for the immersed objects were reported previously (Uhlmann, 2003; Tai et al., 2005; Wang et al., 2008; Blasco et al., 2009; Clausen et al., 2010; Borazjani et al., 2012; Yildirim et al., 2013; Wang et al., 2013). Nevertheless, the time and spatial discretization schemes of our Fictitious Domain (referred to as FD) method are different from those reported, and in particular, our key parallelization strategy for the particle-related problem (i.e., the introduction of main and subordinate particle lists) were not mentioned in previous studies. In addition, our new parallel code is used to investigate the effects of particle inertia on the turbulent channel flow.

The rest of paper is organized as follows. The parallelization algorithms of our fictitious domain method are presented in section 2. The scalability of the parallel code is tested in section 3. In section 4, the new code is applied to the simulation of particle-laden turbulent channel flows with different particle-to-fluid density ratios. Concluding remarks are given in the final section.

2 Numerical method

2.1 Formulation of the FD method

In the FD method, the interior of each solid particle is filled with the fluid and a pseudo body-force is introduced over the particle inner domain to enforce the fictitious fluid to satisfy the rigid-body motion constraint. For simplicity of description, only one particle is considered. Let ρ_s, V_p, **J**, ***U***, and $\boldsymbol{\omega}_s$ represent the particle density, volume, moment of inertia tensor, translational velocity and angular velocity, respectively. Fluid viscosity and density are denoted by μ and ρ_f, respectively. *P(t)* represents the solid domain, and Ω the entire domain comprising both interior and exterior of the body. The governing equations can be non-dimensionalized by introducing the following scales: L_c for length, U_c for velocity, L_c/U_c for time, $\rho_f U_c^2$ for the pressure, and $\rho_f U_c^2/L_c$ for the pseudo body-force. The dimensionless governing equations in strong form for the FD method are (Yu and Shao, 2007):

$$\frac{\partial \mathbf{u}}{\partial t}+\mathbf{u}\cdot\nabla\mathbf{u}=\frac{\nabla^2\mathbf{u}}{\mathrm{Re}}-\nabla p+\boldsymbol{\lambda}, \qquad \text{in } \Omega, \tag{1}$$

$$\mathbf{u}=\mathbf{U}+\boldsymbol{\omega}_s\times\mathbf{r}, \qquad \text{in } P(t), \tag{2}$$

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

$$(\rho_r-1)V_p^*\left(\frac{d\mathbf{U}}{dt}-Fr\frac{\mathbf{g}}{g}\right)=-\int_P \boldsymbol{\lambda}\,d\mathbf{x}, \tag{4}$$

$$(\rho_r-1)\frac{d(\mathbf{J}^*\cdot\boldsymbol{\omega}_s)}{dt}=-\int_P \mathbf{r}\times\boldsymbol{\lambda}\,d\mathbf{x}. \tag{5}$$

In the above equations, **u** represents the fluid velocity, *p* the fluid pressure, **λ** the pseudo body-force (Lagrange multiplier) defined in the solid particle domain, **r** the position vector with respect to the particle mass center, ρ_r the solid-fluid density ratio defined by $\rho_r=\rho_s/\rho_f$, g the gravitational acceleration, *Re* the Reynolds number defined by $Re=\rho_f U_c L_c/\mu$, *Fr* the Froude number defined by $Fr=gL_c/U_c^2$, V_p^* the dimensionless particle volume defined by $V_p^*=V_p/L_c^3$, and $\mathbf{J}^*$ the dimensionless moment of inertia tensor defined by $\mathbf{J}^* = \mathbf{J} / \rho_s L_c^5$.

2.2 Discretization schemes

A fractional-step time integration scheme is used to decouple the combined system (1-5) into two sub-problems.

(1) Fluid sub-problem for $\mathbf{u}^*$ and p^{n+1} :

$$\frac{\mathbf{u}^\# - \mathbf{u}^n}{\Delta t} - \frac{1}{2}\frac{\nabla^2 \mathbf{u}^\#}{Re} = -\nabla p^{n+1} - \frac{1}{2}\left[3\left(\mathbf{u}\cdot\nabla\mathbf{u}\right)^n - \left(\mathbf{u}\cdot\nabla\mathbf{u}\right)^{n-1}\right] + \frac{1}{2}\frac{\nabla^2\mathbf{u}^n}{Re} + \lambda^n \tag{6}$$

$$\nabla\cdot\mathbf{u}^* = 0 \tag{7}$$

The fluid sub-problem (6)-(7) is a Navier-Stokes problem. The projection scheme is used to further decomposed equations (6-7) into three sub-problems:

(a) Helmholtz equation for velocity:

$$\frac{\mathbf{u}^\# - \mathbf{u}^n}{\Delta t} - \frac{1}{2}\frac{\nabla^2 \mathbf{u}^\#}{Re} = -\nabla p^{n} - \frac{1}{2}\left[3\left(\mathbf{u}\cdot\nabla\mathbf{u}\right)^n - \left(\mathbf{u}\cdot\nabla\mathbf{u}\right)^{n-1}\right] + \frac{1}{2}\frac{\nabla^2\mathbf{u}^n}{Re} + \lambda^n \tag{8}$$

(b) Poisson equation for pressure:

$$\nabla^2\phi = \frac{\nabla\cdot\mathbf{u}^\#}{\Delta t} \quad \left(\frac{\partial\phi}{\partial n} = 0 \quad \text{on wall boundaries}\right) \tag{9}$$

(c) Correction of velocity and pressure:

$$\mathbf{u}^* = \mathbf{u}^\# - \Delta t\nabla\phi; \quad p^{n+1} = p^n + \phi \tag{10}$$

The finite-difference scheme on half-staggered grids is employed for spatial discretization. All spatial derivatives are discretized with the second-order central difference scheme.

(2) Particle sub-problem for $\boldsymbol{U}^{n+1}$, $\boldsymbol{\omega}_s^{n+1}$, $\boldsymbol{\lambda}^{n+1}$,$\mathbf{u}^{n+1}$:

$$\rho_r V_p^* \frac{\mathbf{U}^{n+1}}{\Delta t} = (\rho_r - 1)V_p^*\left(\frac{\mathbf{U}^n}{\Delta t} + Fr\frac{\mathbf{g}}{g}\right) + \int_P \left(\frac{\mathbf{u}^*}{\Delta t} - \lambda^n\right) d\mathbf{x} \tag{11}$$

$$\rho_r \frac{\mathbf{J}^*\cdot\boldsymbol{\omega}_s^{n+1}}{\Delta t} = (\rho_r - 1)\left[\frac{\mathbf{J}^*\cdot\boldsymbol{\omega}_s^n}{\Delta t} - \boldsymbol{\omega}_s^n \times (\mathbf{J}^*\cdot\boldsymbol{\omega}_s^n)\right] + \int_P r\times\left(\frac{\mathbf{u}^*}{\Delta t} - \boldsymbol{\lambda}^n\right) d\mathbf{x} \tag{12}$$

In the above equations (11-12), all the right-hand side terms are known quantities, so $\boldsymbol{U}^{n+1}$ and $\boldsymbol{\omega}_s^{n+1}$ are obtained explicitly. Then, $\boldsymbol{\lambda}^{n+1}$ defined at the Lagrangian nodes can be determined as follows:

$$\boldsymbol{\lambda}^{n+1} = \frac{\mathbf{U}^{n+1} + \boldsymbol{\omega}_s^{n+1}\times\mathbf{r} - \mathbf{u}^*}{\Delta t} + \boldsymbol{\lambda}^n, \tag{13}$$

Finally, the fluid velocities $\mathbf{u}^{n+1}$ at the Eulerian nodes are corrected:

$$\mathbf{u}^{n+1} = \mathbf{u}^* + \Delta t(\boldsymbol{\lambda}^{n+1} - \boldsymbol{\lambda}^n). \tag{14}$$

In above equations, the quantities with the superscript 'n' mean their values at the time of $n\Delta t$, i.e., at the nth time-level, whereas $\mathbf{u}^*$ and $\mathbf{u}^\#$ represent the fluid velocities at the fractional time steps from the nth to (n+1)th time levels.

We choose the tri-linear function, as discrete delta function, to transfer the quantities between the Lagrangian and Eulerian nodes (see reference Yu and Shao, 2007). Only spherical particles are considered in the present study. The position of the particle mass center can be determined from the kinematic equation $\frac{d\mathbf{X}}{dt} = \mathbf{U}$. The reader is referred to Yu and Shao (2007) for more details on our FD method.

2.3 Parallel algorithms

Since the computational domain is rectangular or can be extended to be rectangular with the fictitious domain technique, it is natural to take the domain decomposition as the parallel strategy: one process copes with one sub-domain. MPI (message passing interface) is adopted to transfer data between sub-domains.

2.3.1 Parallelization strategy for the fluid sub-problem

In the fluid sub-problem, the main tasks are to solve the velocity Helmholtz equation (8) and the pressure Poisson equation (9). In our serial code, the Helmholtz equation is discretized into a tri-diagonal algebraic system by using the ADI (Alternating direction implicit) scheme, and the Poisson equation for the pressure is solved with a combination of a fast cosine transformation (FCT) and a tri-diagonal system solver (Yu and Shao, 2007). For our parallel code, we employ the multi-grid iterative method to solve these two equations, considering that it is more convenient to parallelize the multi-grid solver than the combination of FCT and the tri-diagonal system solver. The multi-grid algorithm mainly includes (Wesseling, 1992):

1) Gauss-Seidel relaxation with red-black ordering for smoothing of residuals;
2) Full weighting residuals for restriction from fine to coarse grids, and linear interpolation of residuals for prolongation from coarse to fine grids.

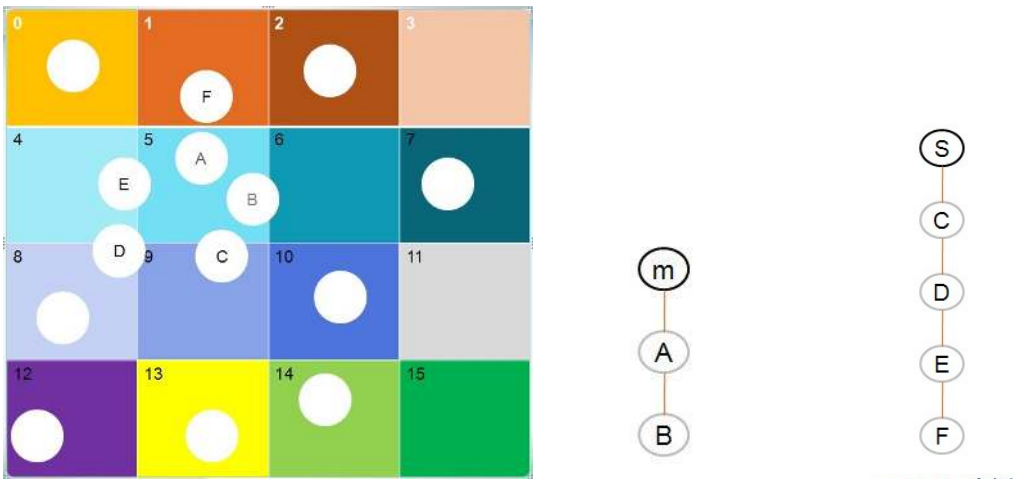
Note that the fluid velocity is defined on grid points, whereas the pressure is defined on the cell center points, therefore, the restriction and interpolation between fine and coarse grids for the fluid velocity and pressure are different (Wesseling, 1992). The reader is referred to Wesseling (1992) for the detailed description of the multi-grid method. For parallelization, the data of each sub-domain are sent to its 26 neighboring sub-domains in terms of point, line and face forms, respectively, depending on the demand from the neighboring sub-domains.

2.3.2 Parallelization strategy for the particle sub-problem

Since the pseudo body-force λ is defined on the Lagrangian nodes and the fluid velocity u defined on the Eulerian nodes, mutual interpolations between these two types of nodes are required. In equations (8) and (14), we need to transfer (or distribute) the pseudo body-force λ from the Lagrangian nodes to the Eulerian nodes, and in equations (11)-(13), transfer the fluid velocity u^* from the Eulerian nodes to the Lagrangian nodes. In the present work, we choose the tri-linear function to transfer the quantities between the Lagrangian and Eulerian nodes. When a particle is crossing the boundary of a sub-domain (e.g., particle B in Figs. 1 and 2), its Lagrangian points are located in different sub-domains. Therefore data communication between sub-domains is needed for the mutual interpolation between the Lagragian and Eulerian points.

The key of our algorithm is to establish two particle lists for each sub-domain: a main list (MP_LIST) and a subordinate list (SP_LIST). The main list is used to update the particle data in simulations and the subordinate list for the data communication in mutual interpolations. The MP_LIST contains the data of particles whose central positions are located in the current sub-domain, and the SP_LIST stores the information of particles whose central positions are not located in the current sub-domain but whose pseudo body-forces have influence on (or are affected by) the velocities in the sub-domain due to the mutual interpolations. For the sub-domain 5 in Fig 2, as an example, the particles A and B belong to MP_LIST because of their central positions located in the sub-domain 5. The particles C, D, E and F belong to SP_LIST. Note that the particle F belongs to SP_LIST because we assume that the gap distance between the particle and the boundary of the sub-domain 5 is smaller than the size of one mesh (h) and thus its pseudo body-force affects (or is affected by) the velocity in the sub-domain 5 (or on the boundary of the sub-domain 5) via the interpolation, although none of the particle domain overlaps with the sub-domain 5. The critical gap distance to determine whether the particle F belongs to SP_LIST depends on the type of the interpolation function (or discrete delta function), and is equal to the mesh size h for the tri-linear function we employed. The particle F is detected by the processor of the sub-domain 1 as the member of its MP_LIST and the member of SP_LIST of the sub-domain 5 according to the particle position. Sub-domain 5 finds this particle by receiving the information from sub-domain 1.

For both MP_LIST and SP_LIST, we establish a new data structure to store the information of particles, including the particle sequence number, position, translational and angular velocities, pseudo body-force and fluid velocity defined on the Lagrangian nodes. For a particle in SP_LIST of each sub-domain, the fluid velocities on the Lagrangian nodes are computed from the interpolation of the velocities on the Eulerian points in the local sub-domain, and thus only the values at part of Lagrangian points are obtained (e.g., the yellow Lagrangian points of particle B in the sub-domain 6 in Fig. 2). For the particle B in Fig. 2, these fluid velocities on the yellow Lagrangian points in SP_LIST of sub-domain 6 are just needed by MP_LIST of sub-domain 5 for a complete set of data. In order to transfer the data in SP_LIST in a sub-domain to MP_LIST in the neighboring sub-domain, we establish an index list for SP_LIST, whose contents include the sequence number of the sub-domain in which the particle center is located (i.e., identifying which MP_LIST), the index (sequence number) of the particle in the identified MP_LIST, and the flags of Lagrangian nodes (being 1 if the fluid velocity at this Lagrangian node is computed in this sub-domain, otherwise 0).



(a) Particle distribution (b) MP_LIST (c) SP_LIST

Figure 1: MP_LIST and SP_LIST for the sub-domain 5 in two dimensions

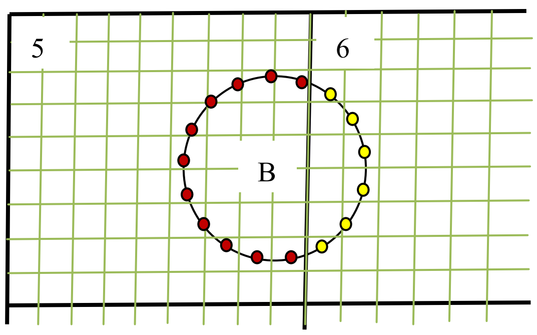


Figure 2: Schematic diagram of the Lagrangian points (marked in yellow) on which the interpolated fluid velocities in SP_LIST of sub-domain 6 are sent to MP_LIST of sub-domain 5.

The full algorithm of our parallel fictitious domain method is the following:

(I) Code initialization, including construction of MP_LIST for each sub-domain with a new data structure.

(II) Establish SP_LIST (as well as the index list) by exchanging the particle data in MP_LIST among sub-domains. For a particle in MP_LIST of one sub-domain, the particle data will be sent to the neighboring sub-domain if it is judged to belong to SP_LIST of the neighboring sub-domain from the distance between the particle surface and the boundary of neighboring sub-domain for the spherical particle considered.

(III) Distribute the pseudo body-force λ from the Lagrangian points to the Eulerian nodes for both MP_LIST and SP_LIST for each local sub-domain. No data communication is needed.

(IV) Solve the fluid sub-problem Eqs. (8)-(10) with the parallel multi-grid method mentioned earlier.

(V) Compute the fluid velocity u^* on the Lagrangian points in Eqs. (11)-(13) from the interpolation of the fluid velocities on the Eulerian nodes for MP_LIST and SP_LIST of each sub-domain (as well as the flags of the Lagrangian points in the index list), and then send the data in SP_LIST to MP_LIST of neighboring sub-domains with the index list, as described earlier.

(VI) Calculate the translational velocity $\boldsymbol{U}^{n+1}$, angular velocity $\boldsymbol{\omega}_s^{n+1}$ and pseudo body-force λ^{n+1} of the particle in MP_LIST for each sub-domain from the equations (11)-(13).

(VII) Correct the fluid velocity $\mathbf{u}^{n+1}$ from Eq. (14). The manipulation is same as step (III).

(VIII) Deal with the collision of the particles with the collision model described below, if needed, otherwise, directly update the particle positions in MP_LIST with the velocities obtained.

(IX) Update MP_LIST. When the particle's central position changes from one sub-domain to another sub-domain, the particle is added to MP_LIST of new sub-domain, and deleted from MP_LIST of old sub-domain.

(X) Delete SP_LIST and the index list, and go back to step (II) for the next time step.

A flowchart for our parallel fictitious domain method, corresponding to the above algorithms, is presented in Fig. 3.

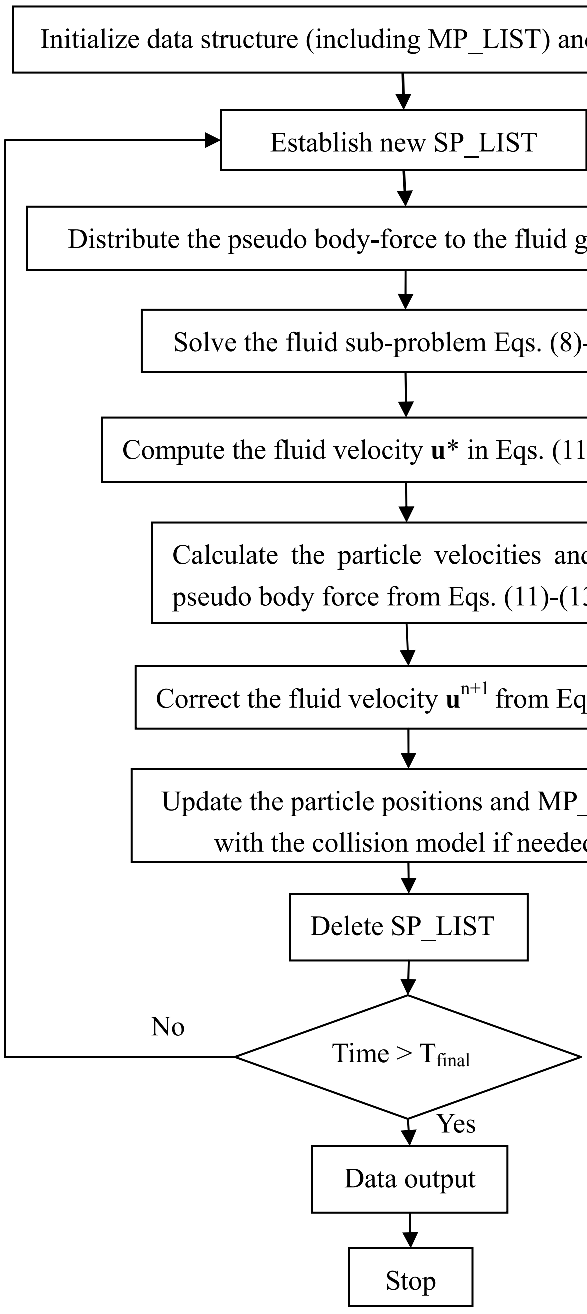

Initialize data structure (including MP_LIST) and set parameters
Establish new SP_LIST
Distribute the pseudo body-force to the fluid grids in Eq. (8)
Solve the fluid sub-problem Eqs. (8)-(10)
Compute the fluid velocity u* in Eqs. (11)-(13)
Calculate the particle velocities and the pseudo body force from Eqs. (11)-(13)
Correct the fluid velocity u^{n+1} from Eq. (14)
Update the particle positions and MP_LIST, with the collision model if needed.
Delete SP_LIST
No
Time > T_final
Yes
Data output
Stop

Figure 3: Flowchart for the parallel fictitious domain method

2.3.3 Parallel implementation strategy for collision model

Next, we describe our parallel algorithm for the particle collision. The soft-sphere collision model is chosen, and the artificial repulsive force is given as:

$$\mathbf{F} = F_0(1 - d / d_c)\vec{\mathbf{n}} \qquad (15)$$

Do you assume d<d_c here?

where F_0 is a constant, d is the particle inter-distance, d_c is a cut-off distance, and $\vec{\mathbf{n}}$ is a unit vector of particle connector. The procedure for our collision model is as follows:

(VIII)-(1) Establish a particle list SP_LIST_C for each sub-domain by gathering the particle data in MP_LIST of neighboring sub-domains.

(VIII)-(2) Seek the particle pairs satisfying the collision criterion, by considering the relative positions (or possibly velocities for hard-sphere model) of the particles in SM_LIST with respect to other particles in SM_LIST (same sub-domain), in SP_LIST_C (neighboring sub-domains) and the wall boundaries.

(VIII)-(3) Calculate the repulsive forces for the detected particle pairs and sum up to obtain the forces on the particles in MP_LIST.

(VIII)-(4) Update the velocities and positions of the particles in MP_LIST in response to the resulting repulsive and resolved hydrodynamic forces.

In addition, we use a small time step (normally one tenth of that for the solution of flow fields) for the collision model, to improve the stability of the explicit scheme for the repulsive force (Glowinski et al., 1999).

3 Scalability of the parallel code

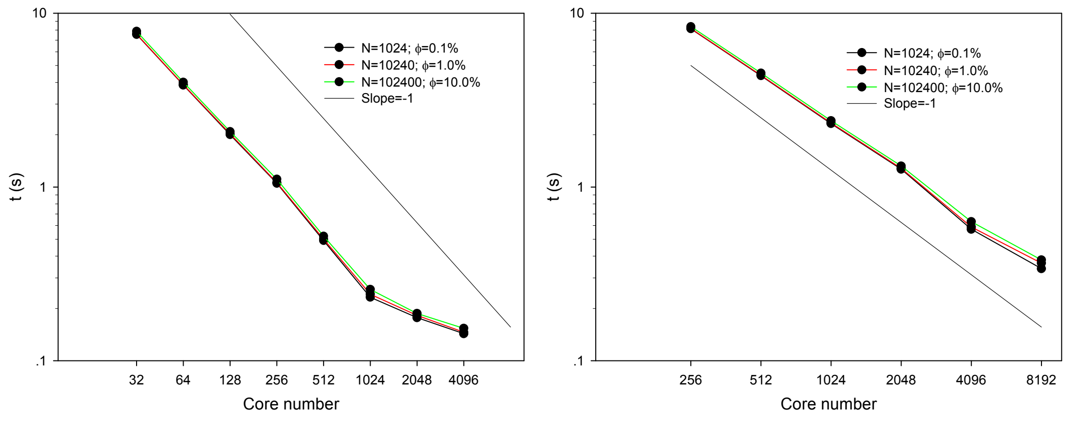


Figure 4: Computational (wall-clock) times per one time step for the particle-laden flows

We choose two cases of particle-laden flows with grid numbers of 512^3 and 1024^3 to test the parallel efficiency of our FD code. The particle numbers are 1024, 10240, and 102400, respectively, corresponding to the volume fractions of 0.1%, 1% and 10%. One particle diameter covers 6.4 grids for the 512^3 grid case and covers 12.8 grids for the 1024^3 grid case. The tests are performed on the Yellowstone supercomputer at the National Center for Atmospheric Research. We run one case for several time-steps and a few times, and then calculate the mean computational (wall-clock) time per time-step. The particle collision and data IO (Input/ Output) are not considered, since we focus on the parallel efficiency of our FD scheme. The particles are initially distributed randomly and homogeneously over the domain. Figure 4 shows the computational time per time-step as a function of core (or processor) number. In the Fig. 4(a), we observe that for the case of 512^3 the computational time is roughly reduced by half when the core number is doubled for the core number up to 1024, which corresponds to the grid number per core of 64×64×32. In other words, the parallel efficiency of our FD code is almost 100%, when the grid number per core does not exceed 64×64×32, above which the efficiency drops due to the increased data communication time relative to the computation time. In addition, Figure 4 shows that the effect of the particle number on the computational time is insignificant for three cases of particle numbers tested; the computational time caused by the presence of the particles is around 10% of the total computational time for the case of 102,400 particles. Figure 4(b) shows results for the case of 1024^3 grid resolution, the speed-up of our parallel code remains largely linear, as the core number is increased up to 8192 (64×64×32 grid number per core).

4 Application example

As mentioned earlier, our parallel FD method differs from our serial FD method only in the solvers for the pressure Poisson equation and the velocity Helmholtz equation. Some tests (such as particle sedimentation and particle inertial migration) show that both versions of our FD method produce almost the same results. Since our serial FD code has been validated in previous studies (Yu and Shao, 2007, 2010), the validations of our new parallel code are not shown here. We apply the new code to the investigation of the particle inertial effects on the turbulent channel flow. Our method represents a direct numerical simulation approach for both particle-laden and turbulent flows. Such methods have been applied to the simulations of particle-laden isotropic turbulent flows (Ten Cate et al., 2004; Lucci et al., 2010; Gao et al., 2013) and turbulent channel flows (Pan and Banerjee, 1997; Uhlmann, 2008; Shao et al., 2012; Kidanemariam et al., 2013; Do-Quang et al., 2014), but not for the effects of the particle inertia (i.e. particle-fluid density ratio) on the turbulent channel flow. In the following, we introduce the numerical model for the problem of interest, validate the accuracy for the single-phase case, conduct the mesh-convergence test for the particle-laden case, and finally report the main results.

4.1 The numerical model

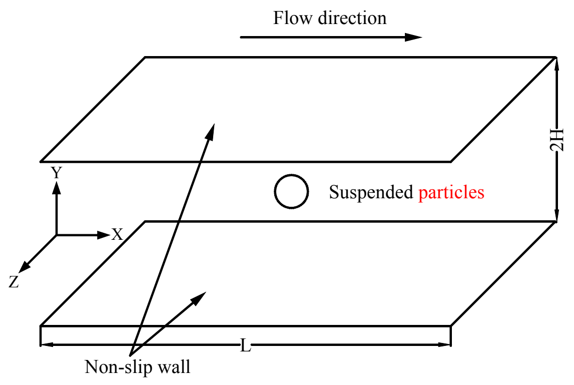


Figure 4: Schematic diagram of particle-laden channel flow

Figure 4 shows the geometrical model of our particle-laden channel flow. Periodic boundary conditions are imposed in the streamwise and spanwise directions, and non-slip wall boundary condition in the transverse direction. Let x, y and z represent respectively the streamwise, transverse and spanwise directions, a the particle radius, and H half of channel height in the transverse direction. We choose H and wall friction velocity $\mathbf{u}_\tau$ as characteristic length and velocity for the non-dimensionlization scheme. The wall friction velocity $\mathbf{u}_\tau$ is defined by the wall shear stress τ and the fluid viscosity ρ_f : $\mathbf{u}_\tau = \sqrt{\tau / \rho_f}$. The friction Reynolds number is defined

[Add density or use kinematic viscosity in Re definition]

by $\mathrm{Re}_\tau = u_\tau H / \mu$. Because of the periodic boundary condition in the streamwise direction, an constant pressure gradient ∇p_e is needed to overcome the wall friction drag to sustain the flow. The value of the constant pressure gradient is $\nabla p_e = -\tau / H$, corresponding to the dimensionless value of $\nabla p_e / (\rho_f u_\tau^2 / H) = -1$.

The friction Reynolds number Re_τ is 180. To study the effects of the particle inertia on the turbulent channel flow, different particle-fluid density ratios of $\rho_r = 1.0,\ 10.42,\ 104.2$ are considered, and the gravity effect is ignored. The particle radius is $a / H = 0.05$. The size of our computational domain is $8H \times 2H \times 4H$. A uniform grid is employed and the grid number is $512 \times 128 \times 256$. The particle volume fraction is 0.84%, corresponding to the particle number of $N_P = 1024$. The dimensionless time step is 0.0001 for $\rho_r = 104.2$ and 0.0002 for the other cases.

4.2 Validation for the single-phase case

We compare our results on the mean and root-mean-square velocities of the particle-free turbulent channel flow to those of Hoyas and Jimenez (2008) who employed the pseudo-spectral method in Fig. 6. One can see that our results for the case of L=8H (i.e. the computational domain size being $8H \times 2H \times 4H$) agree well with those of Hoyas and Jimenez. For the low friction Reynolds number of 180 considered, the sizes of H ×4 2 2 ×H H and H × 2 2 ×H H 2 appear too small to produce resolution-independent results. This is the reason domain size of $8H \times 2H \times 4H$ .why we choose the computational

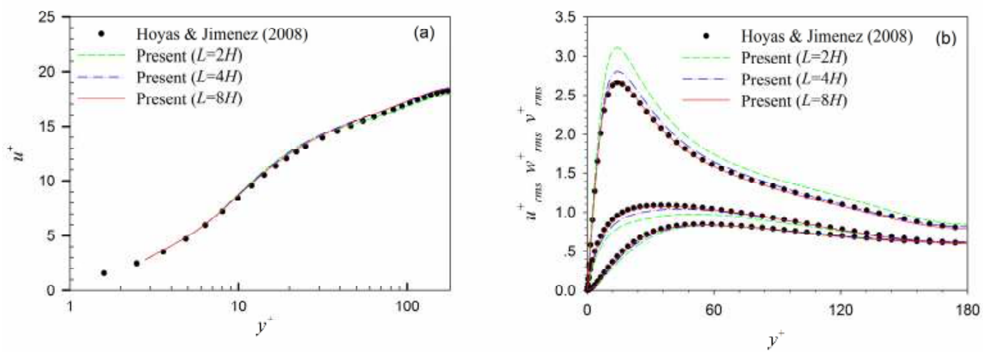


Figure 6: Comparisons of mean and root-mean-square velocities of the particle-free turbulent channel flow.

4.3 Mesh-convergence test for the particle-laden case

For the turbulent particle-laden channel flows, no benchmark data are available to validate the accuracy, and to our knowledge, even no mesh-convergence tests have been performed presumably due to high computational cost. For our simulation case of a H =/ 0 . 0 5 , there are only 3.2 meshes per particle radius, and one may question whether such mesh resolution is high enough to ensure acceptable accuracy. With the parallel code, it is now possible for us to conduct a mesh-convergence test. The parameters for the test case are $a/H = 0.05$ and $\rho_r = 104.2$. The reason why we choose this density ratio is that the RMS (root-mean-square) velocities for $\rho_r = 104.2$ deviate significantly from those for the particle-free case, as shown in Fig. 9. For mesh *h*=*a*/3.2, the grid number is $512\times128\times256$, and the time step is 0.0001, whereas for mesh *h*=*a*/6.2, the grid number is $1024\quad256\times\times512$, and the time step is 0.00005, which is required by the stability condition due to mesh refinement. The results on the RMS velocities for two meshes are plotted in Fig. 7. We observe a good agreement between the two results. The maximum relative error at the peaks of the streamwise RMS velocities is around 3% (2.35 vs. 2.28) in Fig. 7.

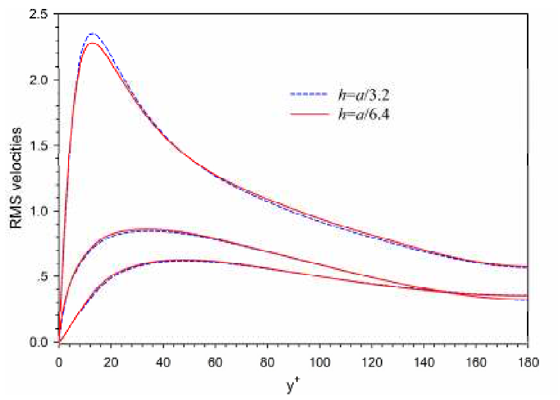


Figure 7: Root-mean-square velocities of the particle-laden turbulent channel flow obtained with two meshes of *h*=*a*/3.2 and *h*=*a*/6.4 in case of $\rho_r = 104.2$.

4.4 Results and discussion

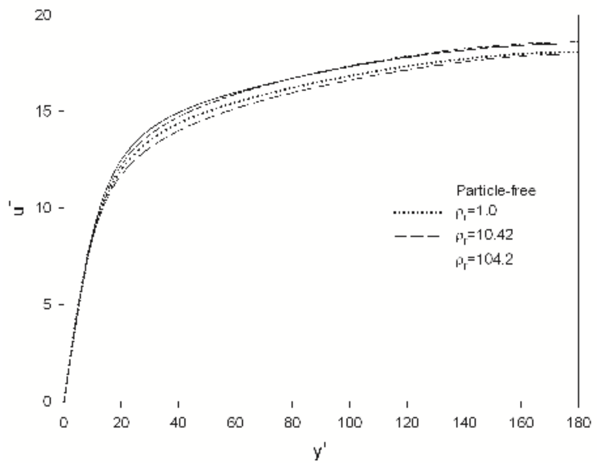


Figure 8: Mean velocity profiles for particle-free and particle-laden turbulent channel flows

Figure 8 shows the mean velocity profiles for all cases studied. One can see that the flow flux first reduces and then increases, as the density ratio increases. Since the pressure gradient is fixed, the reduction in the flow flux means the increase in the flow resistance.

Figure 9 shows the root-mean-square velocities in three directions. Generally, the presence of

particles enhances the RMS velocities near the wall (very close to the wall for the streamwise
remove "wall" here
component and not clearly shown in Fig. 9) and attenuates the RMS velocities at the wall region away from the wall. The suppression of the RMS velocities at the bulk region becomes more pronounced for larger density ratio.

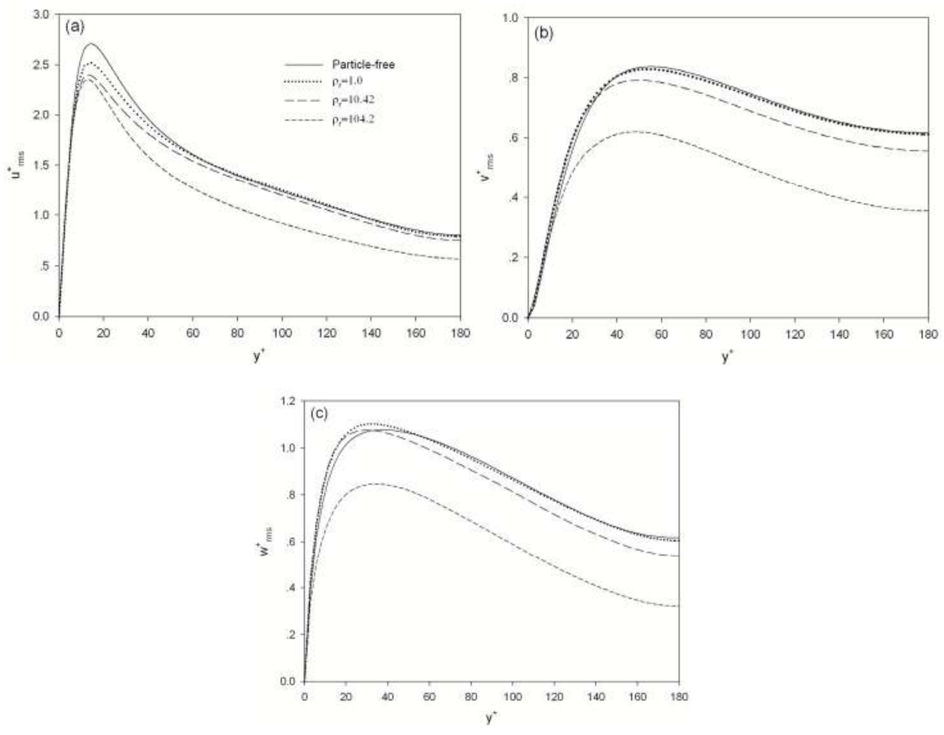
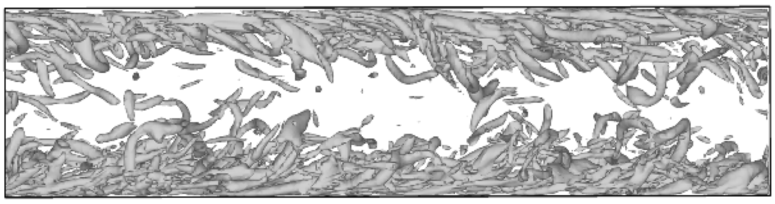


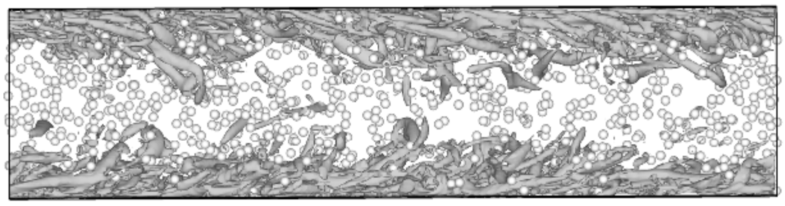
Figure 9: Root-mean-square velocities in the streamvise (a), transverse (b) and spanwise (c) directions for particle-free and particle-laden turbulent channel flows

Figure 10 shows the vortex structures for the single-phase case and the density ratios of 10.42 and 104.2. The presence of particles weakens the large-scale vortices, and the effect is enhanced with increasing density ratio. A significant suppression of the large-scale vortices is found at ρ_r =104.2, which is clearly responsible for the considerable attenuation of the RMS velocities at ρ_r =104.2 in Fig. 9. The presence of particles causes additional viscous dissipation (Lucci et al., 2010), which has dual effects on the flow drag. On one hand, more viscous dissipation means higher viscosity of the suspension and thereby larger flow drag. On the other hand, more viscous dissipation leads to suppression of the large-scale quasi-streamwise vortices which are primarily responsible for the drag-enhancement of turbulence with respect to laminar flow, and thereby lower flow resistance. The competition of these two effects gives rise to the results observed earlier: the flow drag first increases and then decreases with increasing density ratio.

5 Conclusions

We have presented a parallel direct-forcing fictitious domain method (DF/FD) for the simulation of particulate flows, and demonstrated its high parallel efficiency. The new code has been applied to study the effects of particle inertia on the turbulent channel flow. The results show that the large-scale vortices are weakened more severely, and the flow friction drag increases first and then reduces, as particle inertia is increased. We conjecture that the competition of the dual effects of particle-induced viscous dissipation on the flow drag is responsible for the variation of the flow drag with particle inertia. Our results on the particle inertial effects in the present study are preliminary, and we plan to conduct a more extensive study in a future study.


(a) Single-phase


(b) ρ_r=10.42

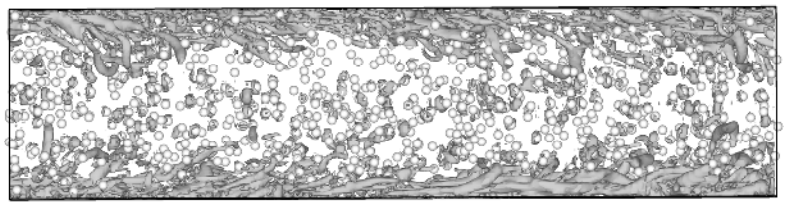

(c) ρ_r=104.2

Figure 10: Vortex structures for (a) the single-phase case, (b) $\rho_r = 10.42$ and (c) $\rho_r = 104.2$.

Acknowledgements

The work was supported by the National Natural Science Foundation of China (Grant Nos 11072217, 11372275 and 11132008), Research Fund for the Doctoral Program of Higher Education of China (20130101110035) and the Program for New Century Excellent Talents in University. LPW is supported by the U.S. National Science Foundation (NSF) under CBET-1235974 and by Air Force Office of Scientific Research under FA9550-13-1-0213. Computing resources at the Yellowstone supercomputer are provided by National Center for Atmospheric Research through CISL-P35751014 and CISL-UDEL0001 and by University of Delaware through NSF CRI 0958512.

References

1 Balachandar S, Eaton JK (2010). Turbulent dispersed multiphase flow. *Annu. Rev. Fluid Mech.* 42:111-133.

2 Blasco J, Calzada MC, Marin M (2009). A fictitious domain, parallel numerical method for rigid particulate flows. *Journal of Computational Physics*; 228: 7596-7613.

3 Borazjani I, Ge L, Le T, Sotiropoulos F (2013). A parallel overset-curvilinear-immersed

boundary framework for simulating complex 3D incompressible flows. *Computers & Fluids* 77: 76-96.

4 Clausen JR, Reasor Jr DA, Aidun CK (2010). Parallel performance of a lattice-Boltzmann/finite element cellular blood flow solver on the IBM Blue Gene/P architecture. *Computer Physics Communications* 181: 1013-1020.

5 Do-Quang M, Amberg G, Brethouwer G, Johansson AV (2014). Simulation of finite-size fibers in turbulent channel flows. *Physical Review E* 89: 013006.

6 Gao H, Li H, Wang LP (2013). Lattice Boltzmann simulation of turbulent flow laden with finite-size particles. *Comput. Math. Appl*. 65: 194-210.

7 Glowinski R, Pan TW, Hesla TI, Joseph DD (1999). A distributed Lagrange multiplier/fictitious domain method for particulate flows. *Int. J. Multiphase Flow* 25:755-794.

8 Hoyas S, Jimenez J (2008). Reynolds number effects on the Reynolds-stress budgets in turbulent channels. *Physics of Fluids* 20: 101511.

9 Hu HH, Patankar A, Zhu MY (2001). Direct numerical simulations of fluid-solid systems using the arbitrary Lagrangian-Eulerian technique. *J. Comput. Phys.* 169: 427-462.

10 Kidanemariam AG, Chan-Braun C, Doychev T, Uhlmann M (2013). Direct numerical simulation of horizontal open channel flow with finite-size, heavy particles at low solid volume fraction. *New Journal of Physics* 15: 025031.

11 Ladd AJC (1994). Numerical simulations of particulate suspensions via a discretized Boltzmann equation. Part 1.Theoretical foundation. *J. Fluid Mech.* 271: 285-309.

12 Lucci F, Ferrante A, Elghobashi S (2010). Modulation of isotropic turbulence by particles of Taylor length-scale size. *J. Fluid Mech*. 650: 5-55.

13 Pan Y, Banerjee S (1997). Numerical investigation of the effects of large particles on wall turbulence. *Physics of Fluids* 9: 3786-3807

14 Shao XM, Yu ZS, Sun B (2008). Inertial migration of spherical particles in circular Poiseuille flow at moderately high Reynolds numbers. *Physics of Fluids* 20: 103307.

15 Shao XM, Wu TH, Yu ZS (2012). Fully resolved numerical simulation of particle-laden turbulent flow in a horizontal channel at a low Reynolds number. *Journal of Fluid Mechanics* 693: 319-344.

16 Tai CH, Zhao Y, Liew KM (2005). Parallel computation of unsteady incompressible viscous flows around moving rigid bodies using an immersed object method with overlapping grids. *Journal of Computational Physics* 207: 151-172.

17 Ten Cate A, Derksen JJ, Portela LM, Van den akker HEA (2004). Fully resolved simulations of colliding monodisperse spheres in forced isotropic turbulence. *J. Fluid Mech*. 519:233-271.

18 Tryggvason G (2010). Virtual motion of real particles. *J. Fluid Mech.* 650: 1-4.

19 Uhlmann M (2003). Simulation of particulate flows on multi-processor machines with distributed memory. *CIEMAT Technical Report* No. 1039, Madrid, Spain.

20 Uhlmann M (2005). An immersed boundary method with direct forcing for the simulation of particulate flows. *J. Comput. Phys.* 209: 448-476.

21 Uhlmann M (2008). Interface-resolved direct numerical simulation of vertical particulate channel flow in the turbulent regime. *Phys. Fluids* 20: 053305.

22 Wang S, He G, Zhang X (2013). Parallel computing strategy for a flow solver based on immersed boundary method and discrete stream-function formulation. *Computers & Fluids* 88: 210-224.

23 Wang ZL, Fan JR, Luo K (2008). Parallel computing strategy for the simulation of particulate flows with immersed boundary method. *Sci China Ser E-Tech Sci* 51: 1169-1176.

24 Wesseling P (1992). An introduction to multigrid methods. John Wiley & Sons, New York, US.

25 Yildirim B, Lin S, Mathur S, Murthy JY (2013). A parallel implementation of fluid–solid interaction solver using an immersed boundary method. *Computers & Fluids* 86: 251-274.

26 Yu ZS, Shao XM (2007). A direct-forcing fictitious domain method for particulate flows. *J. Comput. Phys.* 227: 292-314.

27 Yu ZS, Shao XM, Wachs A, A fictitious domain method for particulate flows with heat transfer. *J. Comput. Phys.* 2006; 217: 424-452.

28 Yu ZS, Phan-Thien N, Tanner RI (2004). Dynamical simulation of sphere motion in a vertical tube. *J. Fluid Mech.* 518: 61-93.

29 Yu ZS, Shao XM (2010). Direct numerical simulation of particulate flows with a fictitious domain method. *Int. J. Multiphase Flow* 36: 127-134.