Fictitious Boundary and Moving Mesh Methods for the Numerical Simulation of Rigid Particulate Flows

Decheng Wan [1] and Stefan Turek [2]

[1] School of Naval Architecture, Ocean and Civil Engineering, Shanghai Jiao Tong University, Huashan Road 1954, 200030 Shanghai, China

[2] Institute of Applied Mathematics LS III, University of Dortmund, Vogelpothsweg 87, 44227 Dortmund, Germany

Abstract

In this paper, we investigate the numerical simulation of rigid particulate flows using a new moving mesh method combined with the multigrid fictitious boundary method (FBM) [9, 10]. With this approach, the mesh is dynamically relocated through a special partial differential equation to capture the region near the surface of the moving particles with high accuracy. The complete system is realized by solving the mesh movement and physical partial differential equations alternately. The flow is computed by an ALE formulation with a multigrid finite element solver, and the solid particles are allowed to move freely through the computational mesh which is adaptively aligned by the moving mesh method based on an arbitrary grid. The important aspect is that the data structures of the underlying undeformed mesh, in many cases a tensorproduct mesh, are preserved while only the spacing between the grid points is adapted in each step. Numerical results demonstrate that the interaction between the fluid and the particles can be accurately and efficiently handled by the presented method. It is also shown that the presented method directly improves upon the previous pure multigrid FBM to solve particulate flows with many moving rigid particles.

Keywords: Particulate Flows, Multigrid, FEM, Fictitious Boundary, Moving Mesh

1 Introduction

Numerical simulation of rigid particulate flows or the motion of small rigid particles in a viscous liquid is one of the main focuses of engineering research and still a challenging task in many applications. Depending on the area of application, these types of problems arise frequently in numerous settings, such as sedimenting and fluidized suspensions, lubricated transport, hydraulic fracturing of reservoirs, slurries, understanding solid-liquid interaction, etc.

Several numerical simulation techniques for particulate flows have been developed over the past decade. In these methods, the fluid flow is governed by the continuity and momentum equations, while the particles are governed by the equation of motion for a rigid body. The flow field around each individual particle is resolved so that the hydrodynamic force acting on the particle is obtained from the fluid solution. Hu, Joseph and coworkers [1, 2], Galdi [3] as well as Maury [4] developed a finite element method based on unstructured grids to simulate the motion of a large number of rigid particles in Newtonian and viscoelastic fluids. This approach is based on an Arbitrary Lagrangian-Eulerian (ALE) technique. Both the fluid and solid equations of motion are incorporated into a single coupled variational equation. The hydrodynamic forces and torques acting on the particles are eliminated in the formulation. The nodes on the particle surface move with the particle, while the nodes in the interior of the fluid are computed using Laplace's equation to guarantee a smoothly varying distribution of nodes. At each time step, a new mesh is generated when the old one becomes too distorted, and the flow field is projected on to the new mesh. In this approach, the positions of the particles and grid nodes are updated explicitly, while the velocities of the fluid and the solid particles are determined implicitly. In the case of 2D, the remeshing of the body-fitted grid can be done by many available grid generation software, but in the more challenging case of a full 3D simulation, the problem of efficient, body-fitted grid generation is not solved in a satisfying manner yet.

Glowinski, Joseph, Patankar and coauthors [5, 6, 7, 8] proposed a distributed Lagrange multiplier (DLM)/fictitious domain method for the direct numerical simulation of large number of rigid particles in fluids. In the DLM method, the entire fluid-particle domain is assumed to be a fluid and then the particle domain is constrained to move with the rigid motion. The fluid-particle motion is treated implicitly using a combined weak formulation in which the mutual forces cancel. This formulation permits the use of a fixed structured grid thus eliminating the need for remeshing the domain. Our group [9, 10, 11, 12] presented another multigrid fictitious boundary method (FBM) for the detailed simulation of particulate flows. The method is based on a fixed unstructured FEM background grid. The motion of the solid particles is modelled by the Newton-Euler equations. Based on the boundary conditions applied at the interface between the particles and the fluid which can be seen as an additional constraint to the governing Navier-Stokes equations, the fluid domain can be extended into the whole domain which covers both fluid and particle domains. The FBM starts with a coarse mesh which may contain already many of the geometrical fine-scale details, and employs a (rough) boundary parametrization which sufficiently describes all large-scale structures with regard to the boundary conditions. Then, all fine-scale features are treated as interior objects such that the corresponding components in all matrices and vectors are unknown degrees of freedom which are implicitly incorporated into all iterative solution steps. An advantage of these fictitious domain methods over the generalized standard Galerkin finite element method is that the fictitious domain methods allow a fixed grid to be used, eliminating the need for remeshing, and they can be handled independently from the flow features. Much progress has been made for adopting the fictitious domain methods to simulate the particulate flows, yet the quest for more accurate and efficient methods remains active. As we know, an underlying problem when adopting the fictitious domain methods is that the boundary approximation is of low accuracy. Particularly in three space dimensions, the ability of the fictitious domain methods to deal accurately with the interaction between fluid and rigid particles is greatly limited. One remedy could be to preserve the mesh topology, for instance as generalized tensorproduct or blockstructured meshes, while a local alignment with the physical boundary of the solid is achieved by a moving mesh process, such that the boundary approximation error can be significantly decreased.

Many authors have recognized that mesh adaption can be an effective tool for simulating sharp fronts or moving interface problems, and reducing numerical dispersion and oscillation. It has been demonstrated that significant improvements in accuracy and efficiency can be gained by adapting the mesh nodes so that they remain concentrated in regions of sharp fronts or interfaces. There are many existing adaptive mesh methods to achieve this purpose. Generally speaking, mesh adaptivity is usually in the form of local mesh refinements or through a continuous mesh mapping. In local adaptive mesh refinement methods [13], an adaptive mesh is obtained by adding or removing points to achieve a desired level of accuracy. This allows a systematic error analysis. However, local refinement methods require complicated data structures and fairly technical methods to communicate information among different levels of refinements. In the mapping approach [14, 15], the mesh points are moved continuously in the whole domain to concentrate in regions where the solution has the largest variations or where the moving interfaces are located. These solution-adaptive or geometry-adaptive meshes maps can be used to compute accurately the sharp variation or the moving interface problems. They also have the additional advantage of allowing the use of standard solvers as all the computations are performed in the logical domain using the undeformed mesh. Over the past decade, several mesh adaptive techniques have been developed, namely the so-called h-, p- and r- methods. The first two do static remeshing with fixed time, where the h-method does automatic refinement or coarsening of the spatial mesh based on a posteriori error estimates or error indicators and the p-method takes higher or lower order approximations locally as needed. In contrast, the r-method (also known as moving mesh method) relocates grid points in a mesh having a fixed number of nodes in such a way that the nodes remain concentrated in regions of rapid variation of the solution or corresponding moving interfaces. The r-method is a dynamic method which means that it uses time stepping or pseudo-time stepping approaches to construct the desired transformation. The r-method or moving mesh method differs from the h- and p-methods in that the former keeps the same number of mesh points throughout the entire solution process, while the later have to treat the tedious hanging node problems. Thus, the size of computation and data structure are fixed, which enables the r-method much easier to incorporate into most CFD codes without the need for the changing of system matrix structures and special interpolation procedures. The

r-method has received more and more attention due to some new developments which clearly demonstrate its potential for problems such as those having moving interfaces [16, 17, 18, 19, 20].

The primary objective of this paper is to combine the multigrid fictitious boundary method (FBM) [9, 10] with the moving mesh method described in [20] for the simulation of particulate flow to check the accuracy of the proposed combining method and to compare its results with the previous pure multigrid fictitious boundary method (FBM). As we have shown in [10], the use of the multigrid FBM does not require to change the mesh during the simulations, although the rigid particles vary their positions. The advantage is that no expensive remeshing has to be performed while a fixed mesh can be used such that in combination with appropriate data structures and fast CFD solvers very high efficiency rates can be reached. However, the accuracy for capturing the surfaces of solid particles is only of first order which might lead to accuracy problems. For a better approximation of the particle surfaces, we adopt a deformed grid, created from a cartesian mesh, in which the topology is preserved, only the grid spacing is changed such that the grid points are concentrated near the surfaces of the rigid particles. Only the solution of additional linear Poisson problems together with simple ODEs in every time step is required for generating the deformation grid, which means that the additional work is significantly less than the main fluid-particle part. The paper is organized as follows. In Section 2, the physical models together with a collision model for the rigid particulate flows are described. The detailed numerical schemes including the multigrid FBM and the moving mesh method are given in Section 3. Extensive numerical experiments and their computational results will be presented in Section 4. Improvements of accuracy over the previous pure multigrid FBM will be emphasized. The concluding remarks will be given in Section 5.

2 Descriptions of the Physical Models

2.1 Governing Equations

In our numerical studies of particle motion in a fluid, we will assume that the fluids are immiscible and Newtonian. The particles are assumed to be rigid. Let us consider the unsteady flow of N particles with mass M_i $(i = 1, \ldots, N)$ in a fluid with density ρ_f and viscosity ν. Denote $\Omega_f(t)$ as the domain occupied by the fluid at time t, and $\Omega_i(t)$ as the domain occupied by the ith particle. So, the motion of an incompressible fluid is governed by the following Navier-Stokes equations in $\Omega_f(t)$,

$$\rho_f \left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} \right) - \nabla \cdot \sigma = 0 \,, \qquad \nabla \cdot \mathbf{u} = 0 \qquad \forall\, t \in (0, T), \tag{1}$$

where σ is the total stress tensor in the fluid phase defined as

$$\sigma = -p\,\mathbf{I} + \mu_f \left[\nabla\mathbf{u} + (\nabla\mathbf{u})^T\right]. \tag{2}$$

Here $\mathbf{I}$ is the identity tensor, $\mu_f = \rho_f \cdot \nu$, p is the pressure and $\mathbf{u}$ is the fluid velocity. Let $\Omega_T = \Omega_f(t) \cup \{\Omega_i(t)\}_{i=1}^N$ be the entire computational domain which shall be independent of time t. Dirichlet- and Neumann-type boundary conditions can be imposed on the outer boundary $\Gamma = \partial\Omega_f(t)$. Since $\Omega_f = \Omega_f(t)$ and $\Omega_i = \Omega_i(t)$ are always depending on t, we drop t in all following notations.

The equations that govern the motion of each particle are the following Newton-Euler equations, i.e., the translational velocities $\mathbf{U}_i$ and angular velocities ω_i of the ith particle satisfy

$$M_i\,\frac{d\,\mathbf{U}_i}{d\,t} = (\Delta M_i)\,\mathbf{g} + \mathbf{F}_i + \mathbf{F}'_i\,, \qquad \mathbf{I}_i\,\frac{d\,\omega_i}{d\,t} + \omega_i \times (\mathbf{I}_i\,\omega_i) = T_i\,, \tag{3}$$

where M_i is the mass of the ith particle; $\mathbf{I}_i$ is the moment of the inertia tensor; ΔM_i is the mass difference between the mass M_i and the mass of the fluid occupying the same volume; $\mathbf{g}$ is the gravity vector; $\mathbf{F}'_i$ are collision forces acting on the ith particle due to other particles which come close to each other. We assume that the particles are smooth without tangential forces of collisions acting on them; the details of the collision model will be discussed in the following subsection. $\mathbf{F}_i$ and T_i are the resultants of the hydrodynamic forces and the torque about the center of mass acting on the ith particle which are calculated by

$$\mathbf{F}_i = (-1)\int_{\partial\Omega_i} \sigma \cdot \mathbf{n}\, d\,\Gamma_i\,, \qquad T_i = (-1)\int_{\partial\Omega_i} (\mathbf{X} - \mathbf{X}_i) \times (\sigma \cdot \mathbf{n})\; d\,\Gamma_i, \tag{4}$$

where σ is the total stress tensor in the fluid phase defined by Eq. (2), $\mathbf{X}_i$ is the position of the mass center of the ith particle, $\partial\Omega_i$ is the boundary of the ith particle, $\mathbf{n}$ is the unit normal vector on the boundary $\partial\Omega_i$ pointing outward to the flow region. The position $\mathbf{X}_i$ of the ith particle and its angle θ_i are obtained by integration of the kinematic equations

$$\frac{d\,\mathbf{X}_i}{d\,t} = \mathbf{U}_i\,, \qquad \frac{d\,\theta_i}{d\,t} = \omega_i. \tag{5}$$

No-slip boundary conditions are applied at the interface $\partial\Omega_i$ between the ith particle and the fluid, i.e., for any $\mathbf{X} \in \bar{\Omega}_i$, the velocity $\mathbf{u}(\mathbf{X})$ is defined by

$$\mathbf{u}(\mathbf{X}) = \mathbf{U}_i + \omega_i \times (\mathbf{X} - \mathbf{X}_i)\,. \tag{6}$$

2.2 Collision Models

For handling more than one particle, a collision model is needed to prevent the particles from interpenetrating each other. Glowinski, Joseph and coauthors [7, 8] proposed repulsive force models in which an artificial short-range repulsive force between particles is introduced for keeping the particle surfaces more than one element (the range of the repulsive force) apart from each other. In these models, overlapping of the regions occupied by the rigid bodies is not allowed since conflicting rigid body motion constraints from two different particles are not imposed at the same velocity nodes. However, in numerical calculations, the overlapping of particles could happen. For solving this problem, Singh, Hesla and Joseph [28] suggested a modified repulsive force model in which the particles are allowed to come arbitrarily close and even to overlap slightly each other. When conflicting rigid body motion constraints from two different particles are applied onto a velocity node, then the constraint from the particle that is closer to that node is used. A repulsive force is only applied when the particles overlap each other.

Following such models, we examine another collision model with a new definition of short range repulsive forces which can not only prevent the particles from getting too close, it can also deal with the case of particles overlapping each other when numerical simulations bring the particles very close due to unavoidable numerical truncation errors. For the particle-particle collisions, the repulsive force is determined as,

$$\mathbf{F}_{i,j}^{P} = \begin{cases} 0, & \text{for } d_{i,j} > R_i + R_j + \rho, \\ \dfrac{1}{\epsilon_P'}(\mathbf{X}_i - \mathbf{X}_j)(R_i + R_j - d_{i,j}), & \text{for } d_{i,j} \le R_i + R_j, \\ \dfrac{1}{\epsilon_P}(\mathbf{X}_i - \mathbf{X}_j)(R_i + R_j + \rho - d_{i,j})^2, & \text{for } R_i + R_j \le d_{i,j} \le R_i + R_j + \rho, \end{cases} \tag{7}$$

where R_i and R_j are the radius of the ith and jth particle, $\mathbf{X}_i$ and $\mathbf{X}_j$ are the coordinates of their mass centers, $d_{i,j} = |\mathbf{X}_i - \mathbf{X}_j|$ is the distance between their mass centers, ρ is the range of the repulsive force (usually $\rho = 0.5 \sim 2.5\Delta h$, Δh is the mesh size), ϵ_P and ϵ_P' are small positive stiffness parameters for particle-particle collisions. If the fluid is sufficiently viscous, and $\rho \simeq \Delta h$ as well as ρ_i/ρ_f are of order 1 (ρ_i is the density of the ith particle, ρ_f is the fluid density), then we can take $\epsilon_P \simeq (\Delta h)^2$ and $\epsilon_P' \simeq \Delta h$ in the calculations. For the particle-wall collisions, the corresponding repulsive force reads,

$$\mathbf{F}_{i}^{W} = \begin{cases} 0, & \text{for } d_i' > 2R_i + \rho, \\ \dfrac{1}{\epsilon_W'}(\mathbf{X}_i - \mathbf{X}_i')(2R_i - d_i'), & \text{for } d_i' \le 2R_i, \\ \dfrac{1}{\epsilon_W}(\mathbf{X}_i - \mathbf{X}_i')(2R_i + \rho - d_i')^2, & \text{for } 2R_i \le d_i' \le 2R_i + \rho, \end{cases} \tag{8}$$

where $\mathbf{X}_i'$ is the coordinate vector of the center of the nearest imaginary particle P_i' located on the boundary wall Γ w.r.t. the ith particle, $d_i' = |\mathbf{X}_i - \mathbf{X}_i'|$ is the distance between the mass centers of the ith particle and the center of the imaginary particle P_i', ϵ_W is a small positive stiffness parameter for particle-wall collisions, usually it can be taken as $\epsilon_W = \epsilon_P/2$ and $\epsilon'_W = \epsilon'_P/2$ in the calculations. Then, the total repulsive forces (i.e. collision forces) exerted on the ith particle by the other particles and the walls can be expressed as follows,

$$\mathbf{F}'_i = \sum_{j=1, j\neq i}^{N} \mathbf{F}^P_{i,j} + \mathbf{F}^W_i \, . \tag{9}$$

3 Numerical Method

3.1 Multigrid FEM Fictitious Boundary Method

The details of multigrid FEM fictitious boundary method have been presented in Refs. [9, 10, 12]. For illustration, a brief description is given below.

The multigrid FEM fictitious boundary method (FBM) is based on a multigrid FEM background grid which covers the whole computational domain Ω_T and can be chosen independently from the particles of arbitrary shape, size and number. It starts with a coarse mesh which may already contain many of the geometrical details of Ω_i ($i = 1, \ldots, N$), and it employs a fictitious boundary indicator (see [9]) which describes all fine-scale structures of the particles with regard to the fluid-particle matching conditions of Eq. (6). Then, all fine-scale features of the particles are treated as interior objects such that the corresponding components in all matrices and vectors are unknown degrees of freedom which are implicitly incorporated into all iterative solution steps (see [12]). Hence, by making use of Eq. (6), we can perform calculations for the fluid in the whole domain Ω_T. The considerable advantage of the multigrid FBM is that the total mixture domain Ω_T does not have to change in time, and can be meshed only once. The domain of definition of the fluid velocity $\mathbf{u}$ is extended according to Eq. (6), which can be seen as an additional constraint to the Navier-Stokes equations (1), i.e.,

$$\begin{cases} \nabla \cdot \mathbf{u} = 0 & (a) \quad \text{for} \;\; \mathbf{X} \in \Omega_T, \\ \rho_f \left(\dfrac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} \right) - \nabla \cdot \sigma = 0 & (b) \quad \text{for} \;\; \mathbf{X} \in \Omega_f, \\ \mathbf{u}(\mathbf{X}) = \mathbf{U}_i + \omega_i \times (\mathbf{X} - \mathbf{X}_i) & (c) \quad \text{for} \;\; \mathbf{X} \in \bar{\Omega}_i, \; i = 1, \ldots, N. \end{cases} \tag{10}$$

For the study of interactions between the fluid and the particles, the calculation of the hydrodynamic forces acting on the moving particles is very important. From Eq. (4), we can see that the surface integrals on the wall surfaces of the particles should be conducted for the calculation of the forces $\mathbf{F}_i$ and T_i. However, in the presented multigrid FBM method, the shapes of the wall surface of the moving particles are implicitly imposed in the fluid field. If we reconstruct the shapes of the wall surface of the particles, it is not only a time consuming work, but also the accuracy is only of first order due to a piecewise constant interpolation from our indicator function. For overcoming this problem, we perform the hydrodynamic force calculations using a volume based integral formulation. To replace the surface integral in Eq. (4) we introduce a function α_i,

$$\alpha_i(\mathbf{X}) = \begin{cases} 1 & \text{for} \quad \mathbf{X} \in \Omega_i, \\ 0 & \text{for} \quad \mathbf{X} \in \Omega_T \setminus \Omega_i, \end{cases} \tag{11}$$

where $\mathbf{X}$ denotes the coordinates. The importance of such a definition can be seen from the fact that the gradient of α_i is zero everywhere except at the wall surface of the ith particle, and equals to the normal vector $\mathbf{n}_i$ of wall surface of the ith particle defined on the grid, i.e., $\mathbf{n}_i = \nabla\alpha_i$. Then, the hydrodynamic forces acting on the ith particle can be computed by

$$\mathbf{F}_i = -\int_{\Omega_T} \sigma \cdot \nabla\alpha_i \, d\Omega, \, , \qquad T_i = -\int_{\Omega_T} (\mathbf{X} - \mathbf{X}_i) \times (\sigma \cdot \nabla\alpha_i) \, d\Omega \, . \tag{12}$$

The integral over each element covering the whole domain Ω_T can be exactly calculated with a standard Gaussian quadrature of sufficiently high order (4×4 points). Since the gradient $\nabla\alpha_i$ is non-zero only near the wall surface of the ith particle, thus the volume integrals need to be computed only in one layer of mesh cells around the ith particle, which leads to a very efficient treatment.

The algorithm of the multigrid FEM fictitious boundary method for solving the coupled system of fluid and particles can be summarized as follows:

1. Given the positions and velocities of the particles, solve the fluid equations Eqs. (10) (a) and (b) in the corresponding fluid domain involving the position of the particles for the fictitious boundary conditions.

2. Calculate the corresponding hydrodynamic forces and the torque acting on the particles by using Eq. (12), and compute the collision forces by Eq. (9).

3. Solve Eq. (3) to get the translational and angular velocities of the particles, and then obtain the new positions and velocities of the particles by Eq. (5).

4. Use Eq. (10) (c) to set the new fluid domain and fictitious boundary conditions, and then advance to solve for the new velocity and pressure of the fluid as described in step 1.

3.2 Moving Mesh Method

In this subsection, we briefly describe the moving mesh method which will be adopted and coupled with the multigrid fictitious boundary method (FBM) to solve numerically for particulate flows. The details of the moving mesh method can be found in Ref. [20].

The moving mesh problem can be equated to constructing a transformation φ, $x = \varphi(\xi)$ from the computational space (with coordinate ξ) to the physical space (with coordinate x). There are two basic types of moving mesh methods, local based and velocity based, generally computing x by minimizing a variational form or computing the mesh velocity $v = x_t$ using a Lagrangian-like formulation. The moving mesh method we will employ belongs to the velocity based method, which is based on Liao's work [16, 17, 18, 19] and Moser's work [21]. This method has several advantages: only linear Laplace problems on fixed meshes are needed to be solved, monitor functions can be obtained directly from error distributions or distance functions, mesh tangling can be prevented, and the data structure is always same as that for the starting mesh.

Suppose $g(x)$ (area function) to be the area distribution on the undeformed mesh, while $f(x)$ (monitor function) in contrast describes the absolute mesh size distribution of the target grid, which is independent of the starting grid and chosen according to the need of physical problems. Then, the transformation φ can be computed via the following four steps:

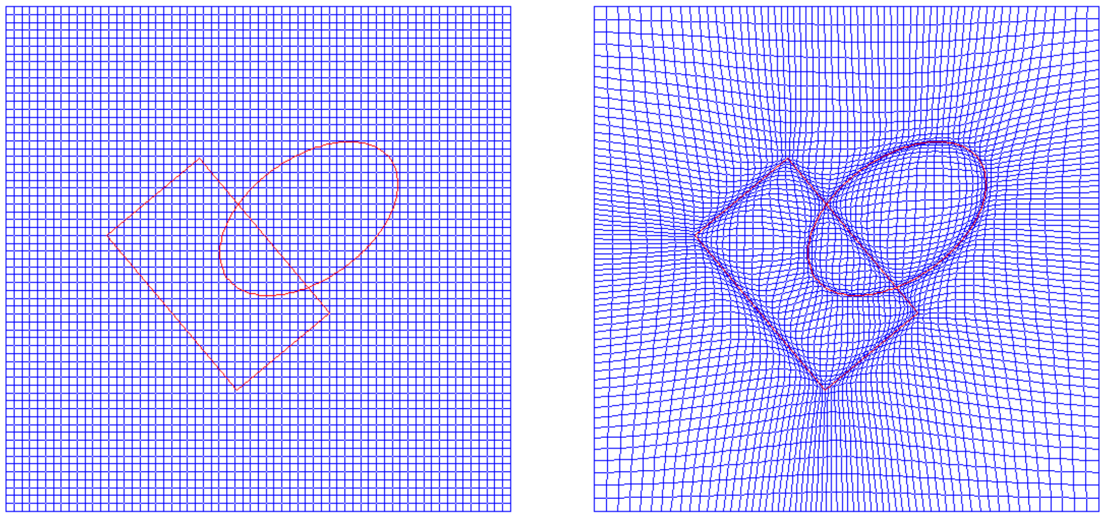
1. Compute the scale factors c_f and c_g for the given monitor function $f(x) > 0$ and the area function g using
$$c_f \int_\Omega \frac{1}{f(x)} dx = c_g \int_\Omega \frac{1}{g(x)} dx = |\Omega|, \tag{13}$$
where $\Omega \subset \mathbb{R}^2$ is a computational domain, and $f(x) \approx$ local mesh area. Let $\tilde{f}$ and $\tilde{g}$ denote the reciprocals of the scaled functions f and g, i.e.,
$$\tilde{f} = \frac{c_f}{f}, \qquad \tilde{g} = \frac{c_g}{g}. \tag{14}$$

2. Compute a grid-velocity vector field $v : \Omega \to \mathbb{R}^n$ by solving the following linear Poisson equation
$$-\text{div}(v(x)) = \tilde{f}(x) - \tilde{g}(x), \;\; x \in \Omega, \quad \text{and} \quad v(x) \cdot \mathfrak{n} = 0, \;\; x \in \partial\Omega, \tag{15}$$
where $\mathfrak{n}$ is the outer normal vector of the domain boundary $\partial\Omega$, which may consist of several boundary components.

3. For each grid point x, solve the following ODE system
$$\frac{\partial \varphi(x,t)}{\partial t} = \eta(\varphi(x,t),t), \quad 0 \le t \le 1, \quad \varphi(x,0) = x, \tag{16}$$
with
$$\eta(y,s) := \frac{v(y)}{s\tilde{f}(y) + (1-s)\tilde{g}(y)}, \quad y \in \Omega, \;\; s \in [0,1]. \tag{17}$$

4. Get the new grid points via
$$\varphi(x) := \varphi(x,1). \tag{18}$$

Here we give two examples for the generation of a deformation mesh using above moving mesh method. Fig. 1 shows the case of two objects with one rectangle and one ellipse inside a square domain. The starting mesh presented in Fig. 1(a) is equidistant, we want to generate a deformation mesh which can align the grid lines near both the surface of the ellipse and the rectangle. We choose the monitor function $f(x)$ as a function of Δd, where Δd is the minimum distance of the grid points to both surfaces of the two solid bodies of the ellipse and the rectangle. Fig. 1(b) gives the generated deformation mesh. We can see that the grid lines are concentrated very well around the surfaces of the two solid bodies.



(a) Equidistant mesh (b) Deformation mesh

Figure 1: First example of a deformation mesh generated from a starting equidistant mesh

Fig. 2 presents another case with two ellipses in a vertical channel. The starting mesh and positions of the two ellipses are given in Fig. 2(a). If we choose the monitor function $f(x)$ as in the above case, i.e. to be the function of Δd, here Δd is the minimum distance of grid points to the both surfaces of the two solid ellipses, then the generated deformation mesh is Fig. 2(b). But we can see that there are still too many grid lines remaining in the lower part of the channel. Moreover, the grid points in the gap between the two ellipses are not distributed very well. Next, we try to use another monitor function $f(x)$, so-called a digit filter function $\Delta d_1 \times \Delta d_2$, here Δd_1 and Δd_2 are the distances of grid points to the surface of No.1 ellipse and No.2 ellipse, respectively, then a much better deformation mesh can be generated. Fig. 2(c) shows the new one. We can see now the grid lines are more concentrated around the surfaces of the two solid ellipses and in the region of the gap between the two ellipses and there are less grid lines staying in the lower part of the channel. It is clearly shown that the quality of the grid distribution is depending on the choice of the monitor function.

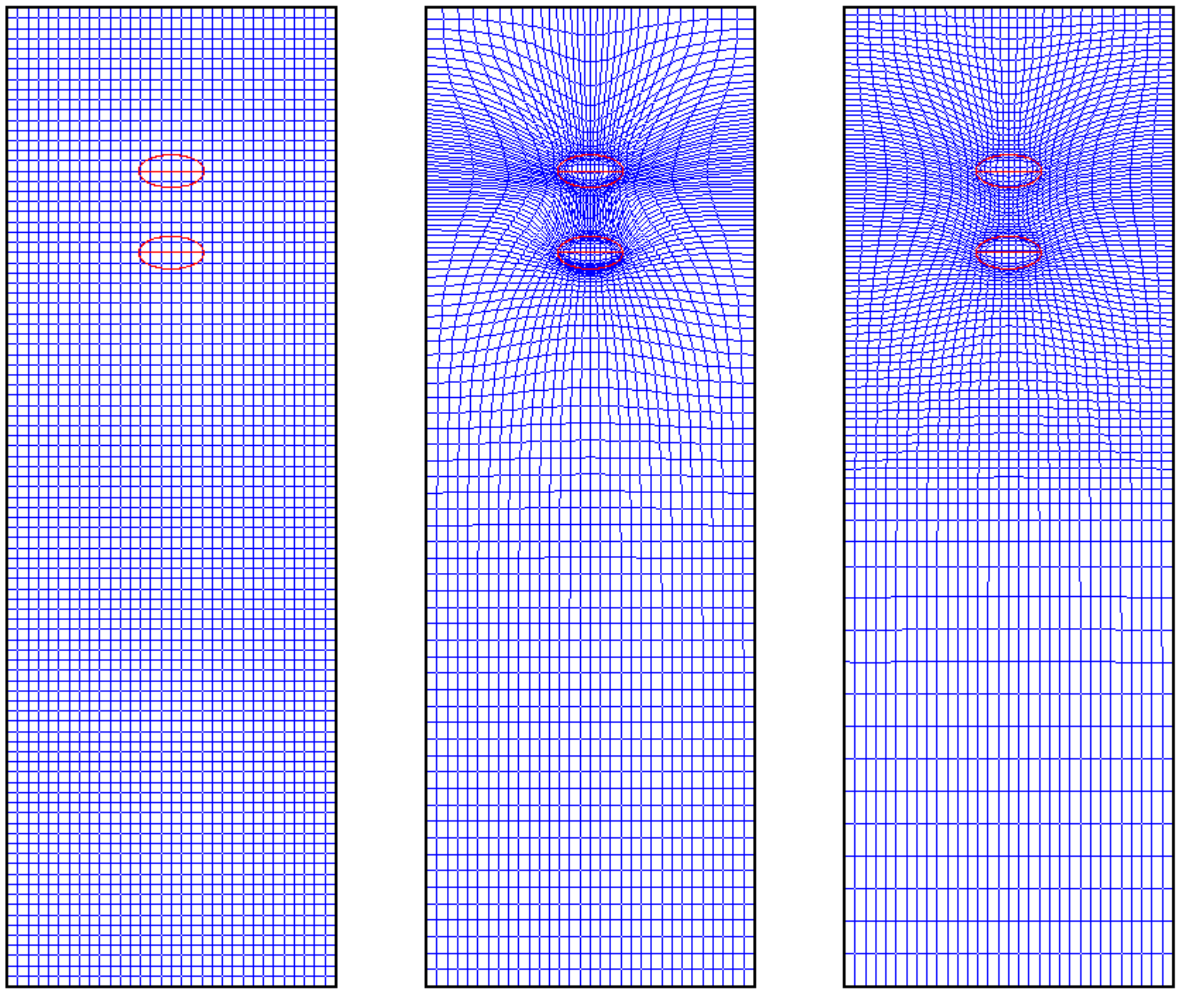

(a) Equidistant mesh (b) Deformation mesh 1 (c) Deformation mesh 2

Figure 2: Second example of a deformation mesh generated from a starting equidistant mesh

3.3 ALE Formulation of the FBM

For a better approximation of the solid particles, we adopt the above described moving mesh method such that we can preserve the mesh topology as generalized tensorproduct or block-structured meshes, while a local alignment with the rigid body surface is reached. The moving mesh method is sometimes referred to as the quasi-Lagrangian method: When the moving mesh method is applied to the multigrid FBM, a mesh velocity $\mathbf{W}_m$ in the convective term in Eq. (10b) must be introduced, i.e.,

$$\rho_f \left[\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} - \mathbf{W}_m) \cdot \nabla \mathbf{u} \right] - \nabla \cdot \sigma = 0 \qquad \text{for} \quad \mathbf{X} \in \Omega_f. \tag{19}$$

In the literature, this is also referred to as an Arbitrary Lagrangian-Eulerian (ALE) formulation. Note that the mesh velocities $\mathbf{W}_m$ do not appear in the continuity equation, as a Pressure-Poisson equation is solved to satisfy the continuity equation in an outer loop. Care has to be taken to satisfy the geometric conservation law (GCL), where the mesh velocity $\mathbf{W}_m$ must be equal to the movement of the mesh velocity $\Delta\mathbf{x}$ during the time step. Therefore, the mesh velocities $\mathbf{W}_m$ should be calculated according to the nodal movement from the previous time step by

$$\mathbf{W}_m = \frac{1}{\Delta t}(\mathbf{x}^{n+1} - \mathbf{x}^n) \tag{20}$$

where Δt is the time step size and n denotes the time step number.

In each time step, a new deformation mesh is generated based on a starting mesh, then the system matrices are updated and the mesh velocity according to the new position of the deformation mesh nodes must be calculated.

3.4 Time Discretization by Fractional-Step-θ Scheme

The fractional-step-θ scheme is a strongly A-stable time stepping approach, it possesses the full smoothing property which is important in the case of rough initial or boundary data. It also contains only very little numerical dissipation which is crucial in the computation of non-enforced temporal oscillations. A more detailed discussion of these aspects can be found in Refs. [22, 24]. We first semi-discretize the Eqs. (10) (a) and (19) in time by the fractional-step-θ scheme. Given $\mathbf{u}^n$ and the time step $K = t_{n+1} - t_n$, then solve for $\mathbf{u} = \mathbf{u}^{n+1}$ and $p = p^{n+1}$. In the fractional-step-θ-scheme, one macro time step $t_n \to t_{n+1} = t_n + K$ is split into three consecutive substeps with $\tilde{\theta} := \alpha\theta K = \beta\theta' K$,

$$\begin{array}{rcl}
[I + \tilde{\theta} N(\mathbf{u}^{n+\theta})]\mathbf{u}^{n+\theta} + \theta K \nabla p^{n+\theta} & = & [I - \beta\theta K N(\mathbf{u}^n)]\mathbf{u}^n \\
\nabla\cdot\mathbf{u}^{n+\theta} & = & 0\,, \\
[I + \tilde{\theta} N(\mathbf{u}^{n+1-\theta})]\mathbf{u}^{n+1-\theta} + \theta' K \nabla p^{n+1-\theta} & = & [I - \alpha\theta' K N(\mathbf{u}^{n+\theta})]\mathbf{u}^{n+\theta} \\
\nabla\cdot\mathbf{u}^{n+1-\theta} & = & 0\,, \\
[I + \tilde{\theta} N(\mathbf{u}^{n+1})]\mathbf{u}^{n+1} + \theta K \nabla p^{n+1} & = & [I - \beta\theta K N(\mathbf{u}^{n+1-\theta})]\mathbf{u}^{n+1-\theta} \\
\nabla\cdot\mathbf{u}^{n+1} & = & 0\,,
\end{array} \tag{21}$$

where $\theta = 1 - \frac{\sqrt{2}}{2}$, $\theta' = 1 - 2\theta$, and $\alpha = \frac{1-2\theta}{1-\theta}$, $\beta = 1 - \alpha$, $N(\mathbf{v})\mathbf{u}$ is a compact form for the diffusive and convective part,

$$N(\mathbf{v})\mathbf{u} := -\nu\, \nabla\cdot\, \left[\nabla\mathbf{u} + (\nabla\mathbf{u})^T\right] + (\mathbf{v} - \mathbf{W}_m)\cdot\nabla\mathbf{u}\,. \tag{22}$$

Therefore, from Eq. (21), in each time step we have to solve nonlinear problems of the following type,

$$[I + \theta_1 K N(\mathbf{u})]\mathbf{u} + \theta_2 K \nabla p = \mathbf{f}\,, \quad \mathbf{f} := [I - \theta_3 K N(\mathbf{u}^n)]\mathbf{u}^n\,, \;\; \nabla\cdot\mathbf{u} = 0\,. \tag{23}$$

For the Eq. (10) (c), we simply take an explicit expression like,

$$\mathbf{u}^{n+1} = \mathbf{U}_i^n + \omega_i^n \times (\mathbf{X}^n - \mathbf{X}_i^n)\,. \tag{24}$$

3.5 Space Discretization by Finite Element Method

If we define a pair $\{\mathbf{u}, p\} \in H := \mathbf{H}_0^1(\Omega) \times L := \mathrm{L}_0^2(\Omega)$, and bilinear forms $a(\mathbf{u}, \mathbf{v}) := (\nabla\mathbf{u}, \nabla\mathbf{v})$ and $b(p, \mathbf{v}) := -(p, \nabla\cdot\mathbf{v})$, a weak formulation of the Eq. (23) reads as follows,

$$\begin{cases} (\mathbf{u}, \mathbf{v}) + \theta_1 K\,[\,a(\mathbf{u}, \mathbf{v}) + n(\mathbf{u}, \mathbf{u}, \mathbf{v})\,] + \theta_2 K\, b\,(p, \mathbf{v}) = (\mathbf{f}, \mathbf{v})\,, & \forall\,\mathbf{v} \in H \\ b\,(q, \mathbf{u}) = 0\,, & \forall\, q \in L \end{cases} \tag{25}$$

here $\mathrm{L}_0^2(\Omega)$ and $\mathbf{H}_0^1(\Omega)$ are the usual Lebesgue and Sobolev spaces, $n(\mathbf{u}, \mathbf{u}, \mathbf{v})$ is a trilinear form defined by

$$n(\mathbf{u}, \mathbf{v}, \mathbf{w}) := \int_\Omega [u_i - (W_m)_i]\left(\frac{\partial v_j}{\partial x_i} + \frac{\partial v_i}{\partial x_j}\right) w_j\, dx\,. \tag{26}$$

To discretize the Eq. (25) in space, we introduce a regular finite-element quadrilateral mesh T_h for the whole computational domain Ω_T, where h is the symbol used as a parameter characterizing the maximum width of the elements of T_h. To obtain the fine mesh T_h from a coarse mesh T_{2h}, we simply connect opposing midpoints. In the fine grid T_h, the old midpoints of the coarse mesh T_{2h} become vertices. We choose the $\tilde{Q}1/Q0$ element pair which uses rotated bilinear shape function for the velocity spanned by $\langle x^2 - y^2, x, y, 1\rangle$ in 2D and piecewise constants for the pressure in cells. The nodal values are the mean values of the velocity vector over the element edges and the mean values of the pressure over the elements rendering this approach nonconforming. The nonconforming $\tilde{Q}1/Q0$ element pair has several important features. It satisfies the Babuška–Brezzi condition without any additional stabilization, and the stability constant is independent of the shape and size of the element. In particular on meshes containing highly stretched and anisotropic cells, the stability and the approximation properties are always satisfied [22]. If we choose finite-dimensional spaces H_h and L_h and define a pair $\{\mathbf{u}_h, p_h\} \in H_h \times L_h$, the discrete problem of Eq. (25) reads,

$$\begin{cases} (\mathbf{u}_h, \mathbf{v}_h) + \theta_1 K\,[\,a_h(\mathbf{u}_h, \mathbf{v}_h) + \tilde{n}_h(\mathbf{u}_h, \mathbf{u}_h, \mathbf{v}_h)\,] & \\ \qquad\qquad + \theta_2 K\, b_h(p_h, \mathbf{v}_h) = (\mathbf{f}, \mathbf{v}_h)\,, & \forall\,\mathbf{v}_h \in H_h \\ b_h(q_h, \mathbf{u}_h) = 0\,, & \forall\, q_h \in L_h \end{cases} \tag{27}$$

where $a_h(\mathbf{u}_h, \mathbf{v}_h) := \sum_{T\in T_h} a(\mathbf{u}_h, \mathbf{v}_h)_{|T}$ and $b_h(p_h, \mathbf{v}_h) := \sum_{T\in T_h} b(p_h, \mathbf{v}_h)_{|T}$. Note that $\tilde{n}_h(\mathbf{u}_h, \mathbf{u}_h, \mathbf{v}_h)$ is a new convective term which includes streamline-diffusion stabilizations defined by

$$\tilde{n}_h(\mathbf{u}_h, \mathbf{v}_h, \mathbf{w}_h) := \sum_{T\in T_h} n(\mathbf{u}_h, \mathbf{v}_h, \mathbf{w}_h)_{|T} + \sum_{T\in T_h} \delta_T(\mathbf{u}_h \cdot \nabla\mathbf{v}_h, \mathbf{u}_h \cdot \nabla\mathbf{w}_h)_{|T}\,, \tag{28}$$

here δ_T is a local artificial viscosity which is a function of a local Reynolds number Re_T,

$$\delta_T := \delta^* \cdot \frac{h_T}{||\mathbf{u}||_\Omega} \cdot \frac{2Re_T}{1 + Re_T}\,, \qquad Re_T = \frac{||\mathbf{u}||_T \cdot h_T}{\nu}\,, \tag{29}$$

where $||\mathbf{u}||_\Omega$ means the maximum norm of velocity in Ω_T, $||\mathbf{u}||_T$ is an averaged norm of velocity over T, h_T denotes the local mesh size of T, and δ^* is an additional free parameter which can be chosen arbitrarily ($\delta^* = 0.1$ is used in our calculations, also see [22]). Obviously, for small local Reynolds numbers, with $Re_T \to 0$, δ_T is decreasing such that we reach in the limit case the standard second order central discretization. Vice versa, for convection dominated flows with $Re_T >> 1$, we add an anisotropic diffusion term of size $O(h)$ which is aligned to the streamline direction $\mathbf{u}_h$.

3.6 Discrete Projection Scheme

For solving the discrete nonlinear problems after time and space discretizations, we have to take the following points into account, i.e., treatment of the nonlinearity, treatment of the incompressibility, and complete outer control like convergence criteria for the overall outer iteration, number of splitting steps, convergence control, embedding into multigrid, etc. In general, there are (at least) two possible approaches for solving the discrete problems [24]. One is the so-called full Galerkin schemes: first, we treat the nonlinearity by an outer nonlinear iteration of fixed point- or quasi-Newton type or by linearization via extrapolation in time, and then we obtain linear subproblems (Oseen equations) which can be solved by a direct coupled or a splitting approach separately for velocity and pressure. Typical schemes are preconditioned GMRES-like or multigrid solvers based on smoothers/preconditioners of type Vanka, SIMPLE or local pressure Schur complement (see [22]). The disadvantage of these approaches is the high numerical cost for small time steps which are typical for particulate flows. Another possibility is the projection type schemes: first we split the coupled problem and obtain definite problems in $\mathbf{u}$ (Burgers equations) as well as in p (Pressure-Poisson problems). Then we treat the nonlinear problems in $\mathbf{u}$ by an appropriate nonlinear iteration or linearization technique while optimal multigrid solvers are used for the Poisson-like problems. Classical schemes belonging to this class are the Chorin and van Kan projection schemes and the discrete projection method, all of them are well suited for dynamic configurations which require small time steps (see [25]).

In this paper, based on the latter approach combined with multigrid methods, we adopt the discrete projection method (DPM) as special variant of the more general multigrid pressure Schur complement (MPSC) schemes to solve the discrete nonlinear problems after time and space discretizations. A detailed description of DPM and MPSC schemes has been presented in [22]: we first perform as outer iteration a fixed point iteration, applied to the fully nonlinear momentum equations. Then, in the inner loop, we solve the corresponding velocity equations involving linear transport-diffusion problems. Finally, the pressure is updated via a Pressure Poisson-like problem, and the corresponding velocity field is adjusted. Since every time step requires the solution of linearized Burgers equations and Poisson-like problems, an optimized multigrid approach is used. The most important components are matrix-vector multiplication, smoothing operator and grid transfer routines (prolongation and restriction) for the underlying FEM spaces which have been realized in FEATFLOW (see [22] for the details).

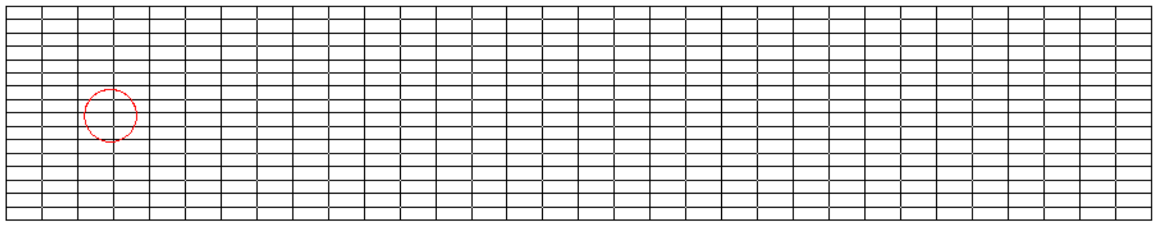
4 Numerical Results

The correct simulation of the proposed method is verified in this section. First of all, a benchmark configuration of 2D flow around a circular body in a channel is given to assess the suitability and accuracy of the hydrodynamic force calculations based on the combination of the multigrid FBM and the moving mesh techniques compared to the results without using the moving mesh techniques. Then, three cases of single moving particle in the fluid are presented to further validate the improvement of accuracy and efficiency through using the presented method. Finally, the drafting, kissing and tumbling of two disks in a channel and the sedimentation of 120 circular particles in a cavity are provided to show the presented method can be easily implemented in the simulation of particulate flows with large number of particles.

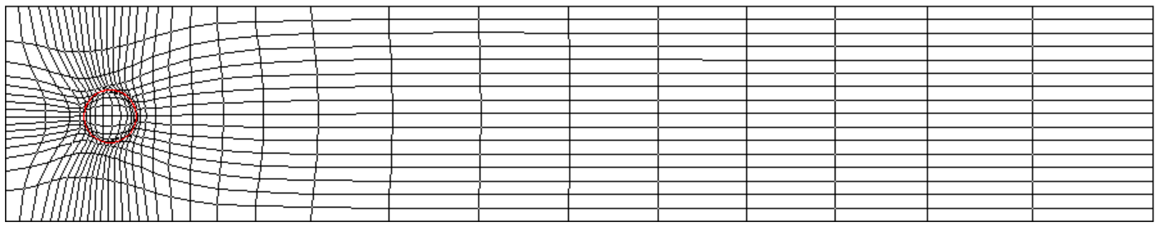
4.1 Benchmark Experiment

We first consider a benchmark case of flow around a fixed circular cylinder in a channel as described in [26]. The channel height is $H = 0.41$, length $L = 2.2$, the cylinder diameter $D = 0.1$. The center point of the cylinder is located at $(0.2, 0.2)$. The Reynolds number is defined by $Re = \bar{U}D/\nu$ with the mean velocity $\bar{U} = 2U(0,\, H/2,\, t)/3$. The kinematic viscosity of the fluid is given by $\nu = \mu_f/\rho_f = 10^{-3}$ and its density $\rho_f = 1$. The inflow profiles are parabolic $U(0,\, Y,\, t) = 6.0\bar{U}Y(H-Y)/H^2$ with $\bar{U} = 0.2$ such that the resulting Reynolds numbers are $Re = 20$ which is a steady case.

Fig. 3 shows the equidistant mesh and deformation mesh employed in our calculations, in which the circle shows the position of the cylinder. The shown meshes can be successively refined by connecting opposite midpoints. The deformation mesh is generated from the equidistant mesh, they both have the same data structure for mesh node and element system, but the deformation mesh has more grid nodes and elements concentrating and aligning around the surface of the cylinder. In Table 1, the parameters for these meshes after several global refinements are given. The meaning of "LEVEL" is the number of refinements, "NVT" the number of vertices, and "NEL" the number of elements.



(a) Equidistant mesh (LEVEL = 4)



(b) Deformation mesh (LEVEL = 4)

Figure 3: Different meshes adopted for flow around a fixed circular cylinder

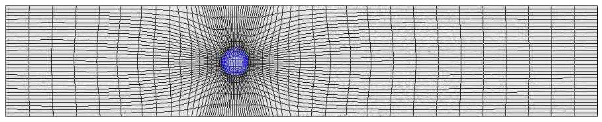
Table 1: Drag and lift values for flow around a fixed circular cylinder with Re = 20

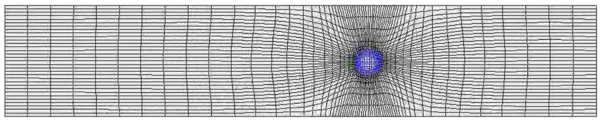
LEVEL			Equidistance mesh		Deformation mesh	
	NVT	NEL	Drag coeff. C_d	Lift coeff. C_l	Drag coeff. C_d	Lift coeff. C_l
4	561	512	4.44688	-0.0649815	6.25486	0.0682610
5	2145	2048	5.31808	-0.3508040	5.72950	0.0297392
6	8385	8192	5.50358	-0.0093675	5.61971	0.0291702
7	33153	32768	5.50585	0.0312388	5.58139	0.0118296
8	131841	131072	5.53049	0.0239737	5.57706	0.0104031
reference values			$C_d = 5.5795$		$C_l = 0.010618$	

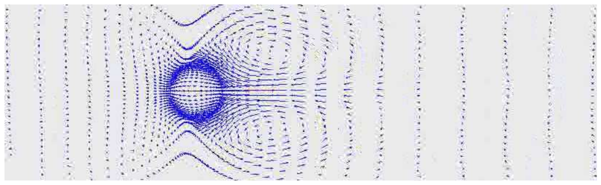
Table 1 presents the computed results of drag and lift coefficients by using the two different meshes. The corresponding reference results for this benchmark problem are also listed in the table for comparison. From the table, we can see that all results are convergent with respect to mesh refinement except that for the lift coefficients when using equidistant meshes. The results for the deformation meshes are much better and more accurate than those for the equidistant meshes. For the deformation mesh cases, already on level 6 with only NEL= 8192, very good and satisfying results have been obtained. Obviously, the accuracy is improved by using the grid deformation techniques.

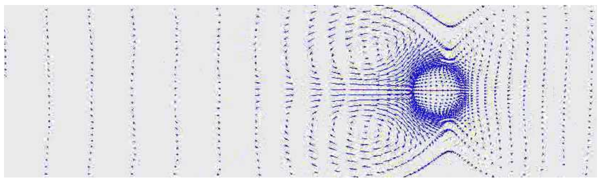
4.2 One Oscillating Cylinder in a Channel

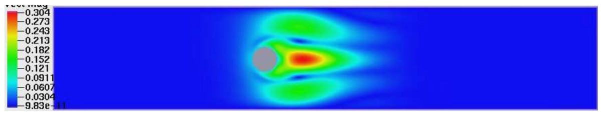
Next, an oscillating circular cylinder in a channel with a prescribed velocity is considered in this numerical test. The channel height is $H = 0.41$, length $L = 2.2$, the cylinder diameter $D = 0.1$. The center point of the cylinder is located initially at $(1.1, 0.2)$. The prescribed velocity for the cylinder is given by $u = 2\pi f\, A \cos(2\pi f\, t)$, $A = 0.25$, $f = 0.25$, $v = 0$, and no-slip velocity conditions are imposed at the two walls, inlet and outlet of the channel. The kinematic viscosity of the fluid is given by $\nu = \mu_f/\rho_f = 10^{-3}$ and its density $\rho_f = 1$. The fluid in the channel is initially at rest.


(a) Deformation mesh ($t = 18.9$)

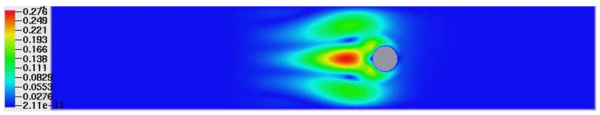

(b) Deformation mesh ($t = 21.0$)


(c) Local vector field ($t = 18.9$)

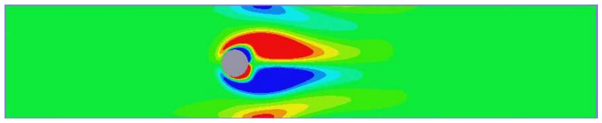

(d) Local vector field ($t = 21.0$)



(e) Norm of velocity ($t = 18.9$)



(f) Norm of Velocity ($t = 21.0$)


(g) Vorticity ($t = 18.9$)

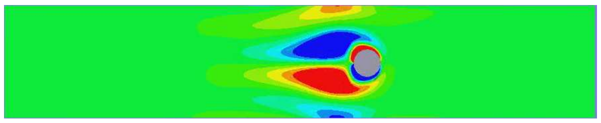

(h) Vorticity ($t = 21.0$)

Figure 4: Oscillating cylinder in a channel

The deformation mesh is generated by using the moving mesh method in each time step in order to always keep grid alignment around the surface of the moving cylinder. Fig. 4 gives two snapshot results at $t = 18.9$ and $t = 21.0$ of the deformation meshes, the local vector field, the norm of velocity and vorticity distribution, respectively. These pictures show that the flow in the channel is disturbed by the oscillating cylinder, and a vortex is generated periodically in the wake of the cylinder. Fig. 5 illustrates the comparison of the computed drag coefficient C_d by the presented method with the reference result based on the body-fitted mesh (see Ref. [12]). The results calculated from LEVEL = 5 to LEVEL = 8 and the parameters of number of elements "NEL" for each refinement mesh are all shown together. From the comparisons, we can see that the presented results are identical with the increase of the mesh refinements, and agreeable very well with the reference result. On the deformation mesh of LEVEL = 5 with 2048 elements, sufficiently good results have already been reached.

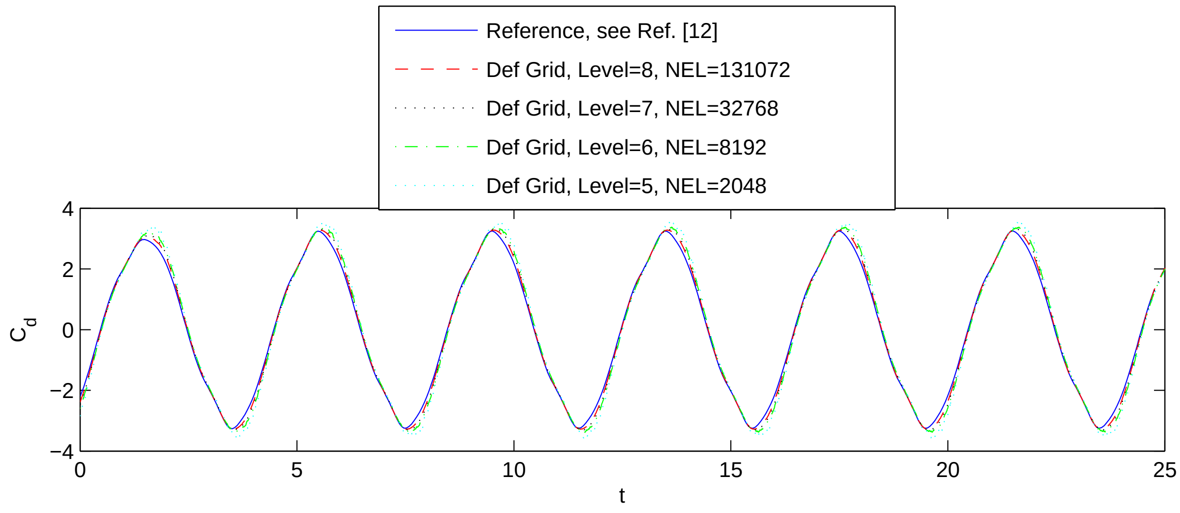
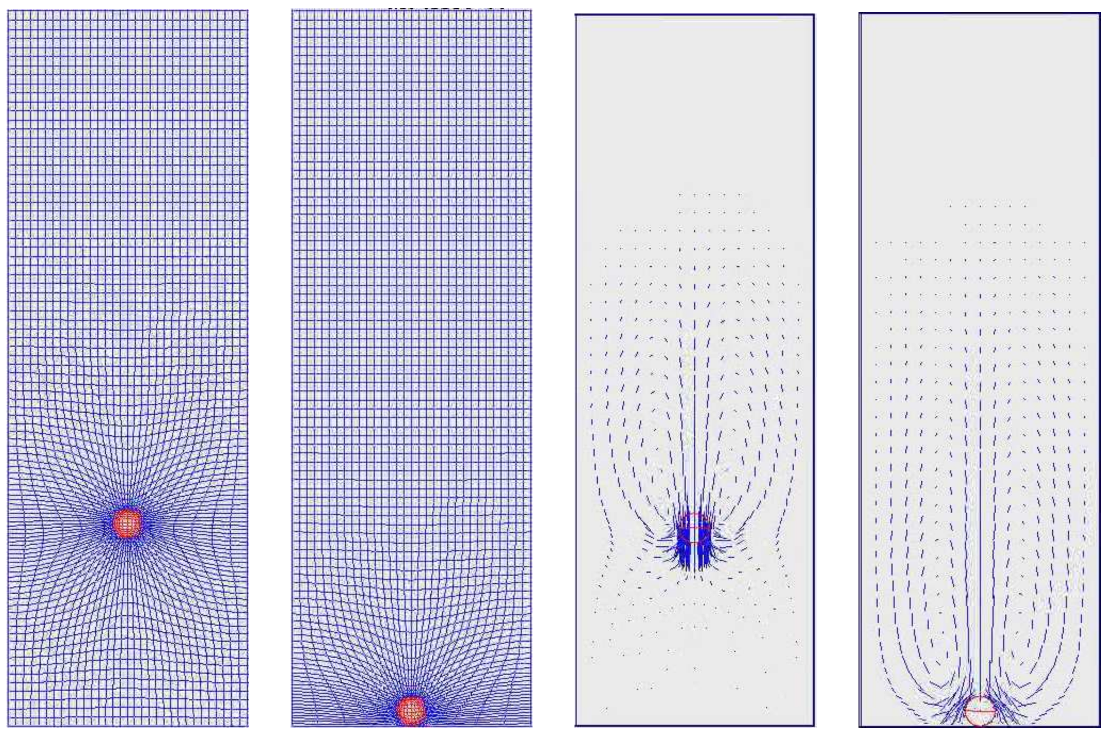


Figure 5: Drag coefficient in time for a oscillating cylinder in a channel

4.3 One 2D Circular Ball Falling Down in a Channel

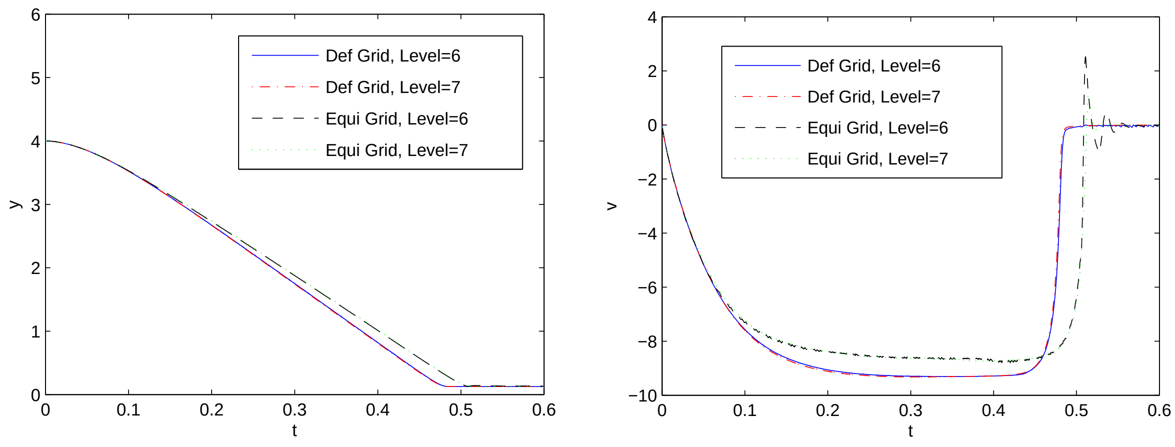
Now we perform a real particle motion as follows: The computational domain is a channel of width 2 and height 6. A rigid circular ball with diameter $d = 0.25$ and density $\rho_p = 1.5$ is located at $(1, 4)$ at time $t = 0$, and it is falling down under gravity in an incompressible fluid with density $\rho_f = 1$ and viscosity $\nu = 0.1$, with gravitational acceleration velocity $g = -980$. We suppose that the ball and the fluid are initially at rest. The simulation is carried out on fixed equidistant meshes and moving deformation meshes, respectively, each of them having two different level, i.e., Level = 6 with 12545 nodes and 12288 elements, as well as Level = 7 with 49665 nodes and 49152 elements.

Fig. 6 gives two results at $t = 0.30$ and $t = 0.48$, showing the deformation mesh and the vector field, respectively. Fig. 7 presents the comparison of the time history of y-coordinate and v-component velocity of the center of the ball by using equidistant meshes and deformation meshes, each of them is calculated by two level meshes LEVEL = 6 and LEVEL = 7, respectively. Both results are convergent with respect to their own mesh refinement. If we compare these results with those obtained by Glowinski in Ref. [8], we can find the results of the deformation meshes are much closer to Glowinski's results than those of the equidistant meshes, which shows again that the accuracy is improved when the moving deformation meshes are employed.



(a) $t = 0.30$ (b) $t = 0.48$ (c) $t = 0.30$ (d) $t = 0.48$

Figure 6: One 2D circular ball falling down in a channel



(a) $y \sim t$ (b) $v \sim t$

Figure 7: The time history of y and v of the center of a 2D ball falling down in a channel

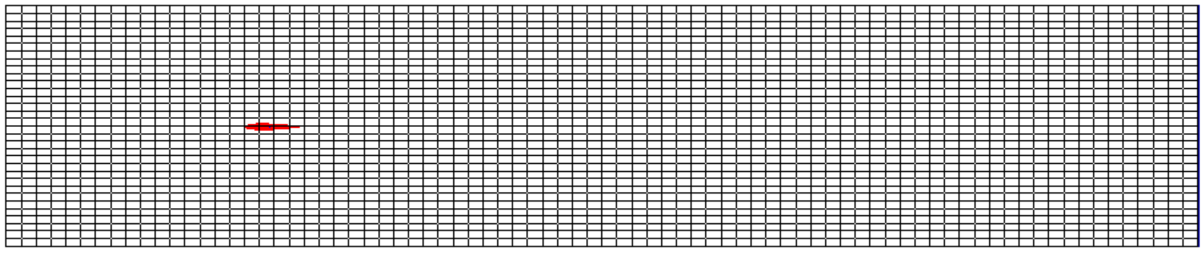
4.4 Induced Rotation of an Airfoil in a Channel

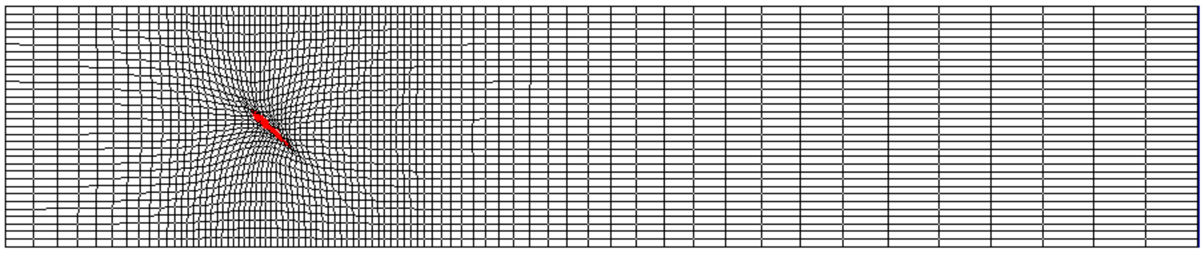
The rigid bodies considered so far have been of circular form. The following simulation will show that the presented method can deal with rigid bodies of more complicated shape very well, and provide better results compared to that without using the moving grid deformation techniques.

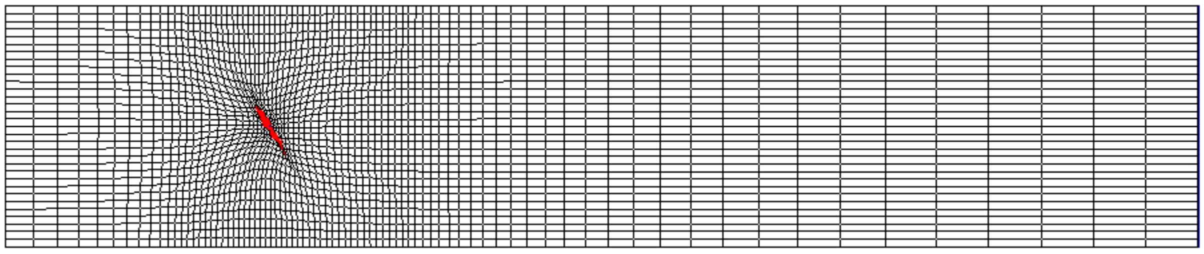
We consider a NACA0012 airfoil that has a fixed center of mass and is induced to rotate freely around its center of mass due to hydrodynamical forces under the action of incoming incompressible viscous flow in a channel. The channel has width 20 and height 4. The density of the fluid is $\rho_f = 1$ and the density of the airfoil is $\rho_p = 1.1$. The viscosity of the fluid is $\nu_f = 10^{-2}$. The initial condition for the fluid velocity is $\mathbf{u} = 0$ and the boundary conditions are given as $\mathbf{u} = 0$ when $y = 0$ or 4 and $\mathbf{u} = 1$ when $x = 0$ or 20 for $t \geq 0$. Initial angular velocity and angle of incidence of the airfoil are zero. The airfoil length is 1.0089304 and the fixed center of mass of the airfoil is at $(0.420516, 2)$. Hence the Reynolds number is about 101 with respect to the length of the airfoil and the inflow speed. The detailed shape of the NACA0012 is described as follows (for $0 \leq X \leq 1.0089304$):

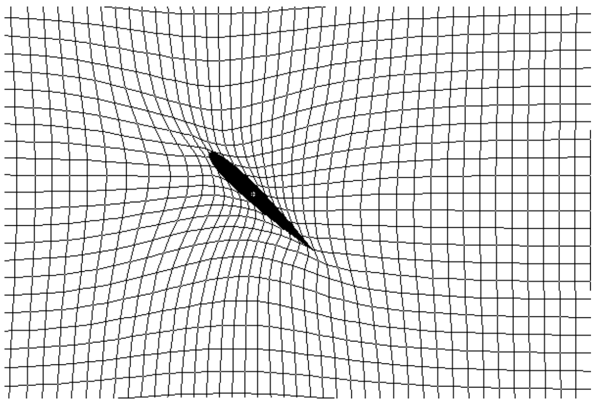
$$Y = 0.6 \cdot \{\, 0.2969 \cdot \sqrt{X} + X \cdot [\, -0.126 + X \cdot [\, -0.3516 + X \cdot (\, 0.2843 - 0.1015 \cdot X\,)\,]\,]\,\}$$

The simulation is implemented on both fixed meshes and moving deformation meshes, each of them having two different levels, i.e., Level = 7 with 41409 nodes and 40960 elements, as well as Level = 8 with 164737 nodes and 163840 elements. The deformation mesh is generated in each time step in order to always keep grid alignment around the surface of the induced rotating airfoil. Fig. 8 (a) shows the starting equidistant mesh used to generate the deformation meshes. In Fig. 8 (b) and (c), two deformation meshes at $t = 14.7$ and $t = 16.0$ are presented, their local zooms are illustrated in Fig. 8 (d) and (e). Fig. 9 plots the time history of the angle of incidence θ and the angular velocity ω of the airfoil calculated by using the equidistant meshes and the deformation meshes, each of them is performed by two levels LEVEL = 7 and LEVEL = 8, respectively. The vector field, vorticity distribution, norm of velocity, and pressure contour for $t = 14.7$ and $t = 16.0$ are given in Fig. 10 and Fig. 11. From these figures and pictures, we can see that the airfoil quickly reaches a periodic motion and tends to keep its broadside perpendicular to the inflow direction which is a stable position for a noncircular rigid body settling in a channel at moderate Reynolds numbers. We observe that the results of the deformation meshes converge better to a mesh independent solution than those of the fixed meshes, and are in excellent agreement with those obtained by Glowinski, Joseph and coauthors [7, 8]. The results of the fixed meshes exhibit much more numerical oscillations and lose stability since they cannot catch very well the velocity field close to the leading edge of the airfoil, which causes the numerical solution to blow up near the leading edge of the airfoil. Obviously, good results and significant accuracy improvement are achieved by using the moving grid deformation techniques. It illustrates that the presented method can easily handle more complex shapes of rigid bodies and obtain more accurate and satisfying results than those without employing the moving grid deformation techniques.


(a) Starting mesh


(b) Deformation mesh ($t = 14.7$)


(c) Deformation mesh ($t = 16.0$)


(d) Zoom mesh ($t = 14.7$)

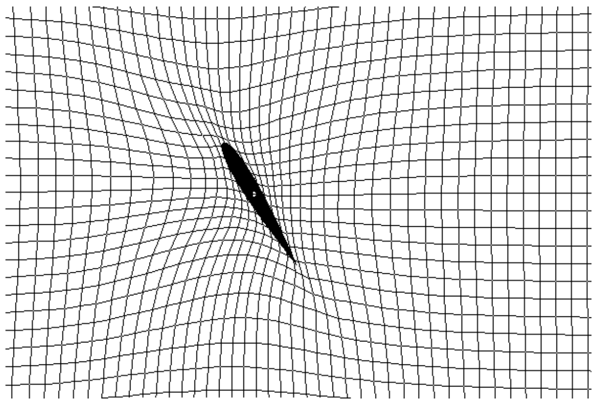
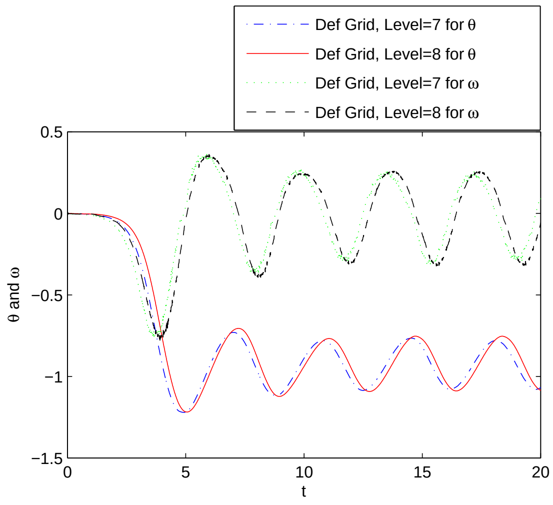
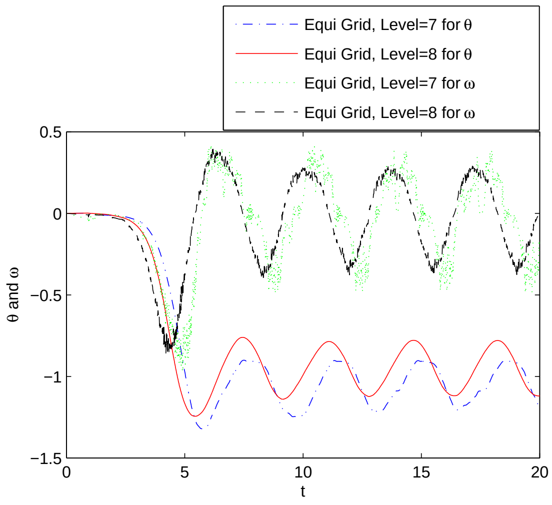

(e) Zoom mesh ($t = 16.0$)

Figure 8: Starting mesh and deformation meshes employed during the simulation of the induced rotation of a NACA0012 airfoil in a channel

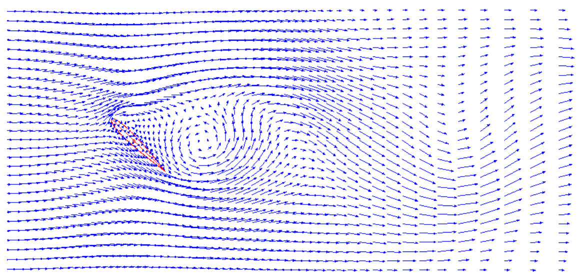


(a) Deformation meshes

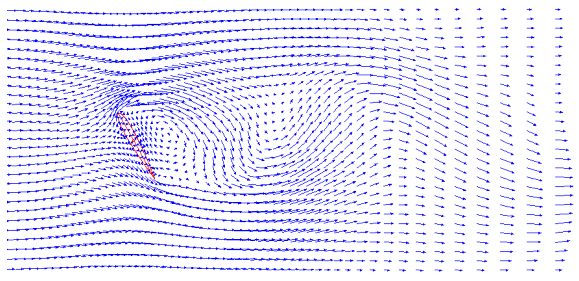


(b) Equidistant meshes

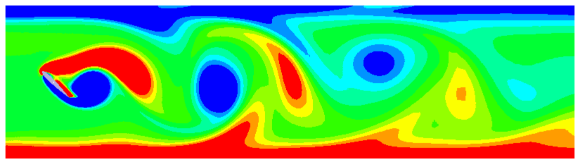
Figure 9: The time history of the angle of incidence θ and the angular velocity ω of the induced rotating NACA0012 airfoil in a channel



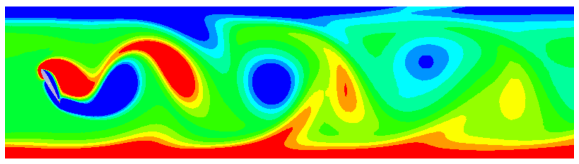
(a) Local vector field ($t = 14.7$)



(b) Local vector field ($t = 16.0$)



(c) Vorticity ($t = 14.7$)

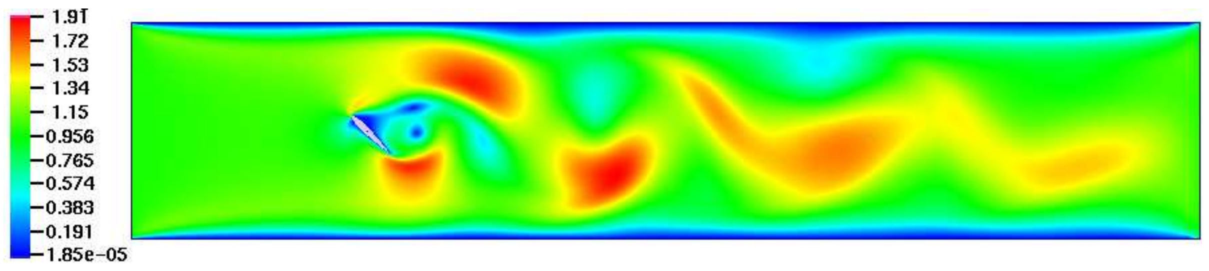


(d) Vorticity ($t = 16.0$)

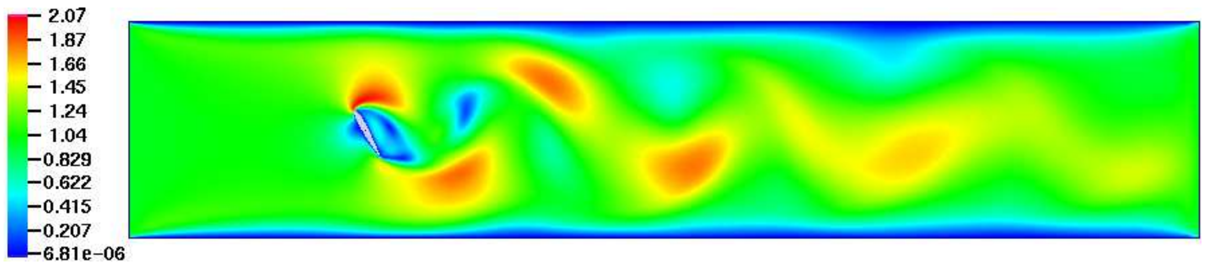
Figure 10: Induced rotation of a NACA0012 airfoil in a channel

4.5 Drafting, Kissing and Tumbling of Two Disks in a Channel

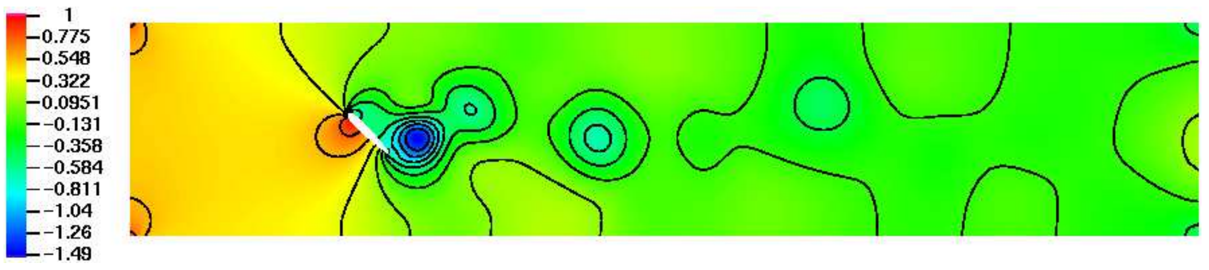
In the following two numerical experiments, we will carry on the cases of multiple particles in a fluid to show that the presented method can be easily implemented for real particulate flow with many moving particles.



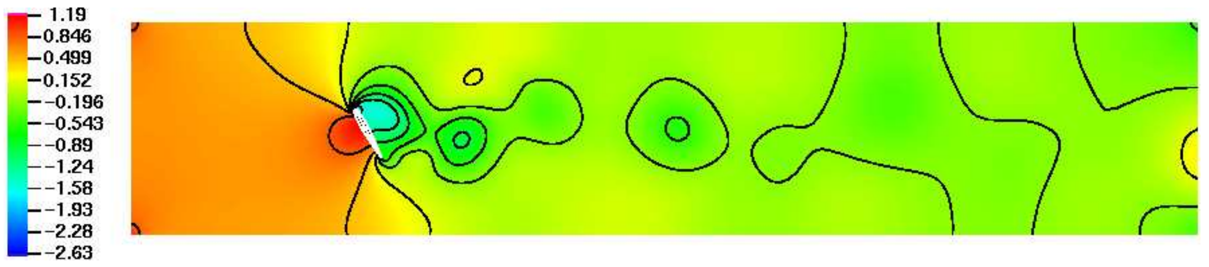
(a) Norm of velocity ($t = 14.7$)



(b) Norm of velocity ($t = 16.0$)



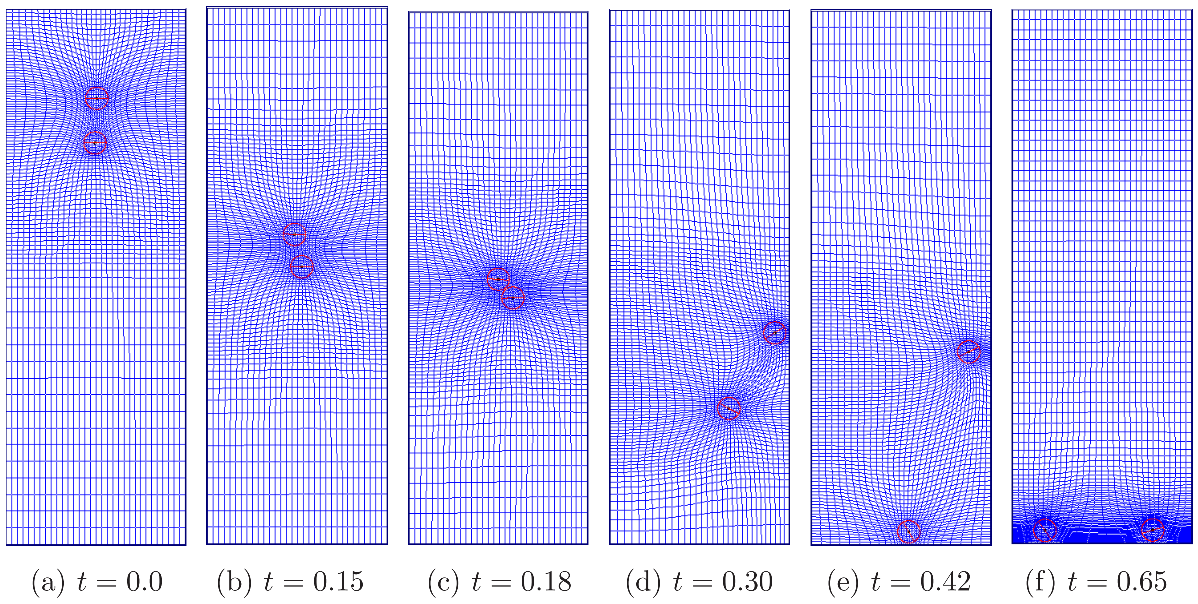
(c) Pressure ($t = 14.7$)



(d) Pressure ($t = 16.0$)

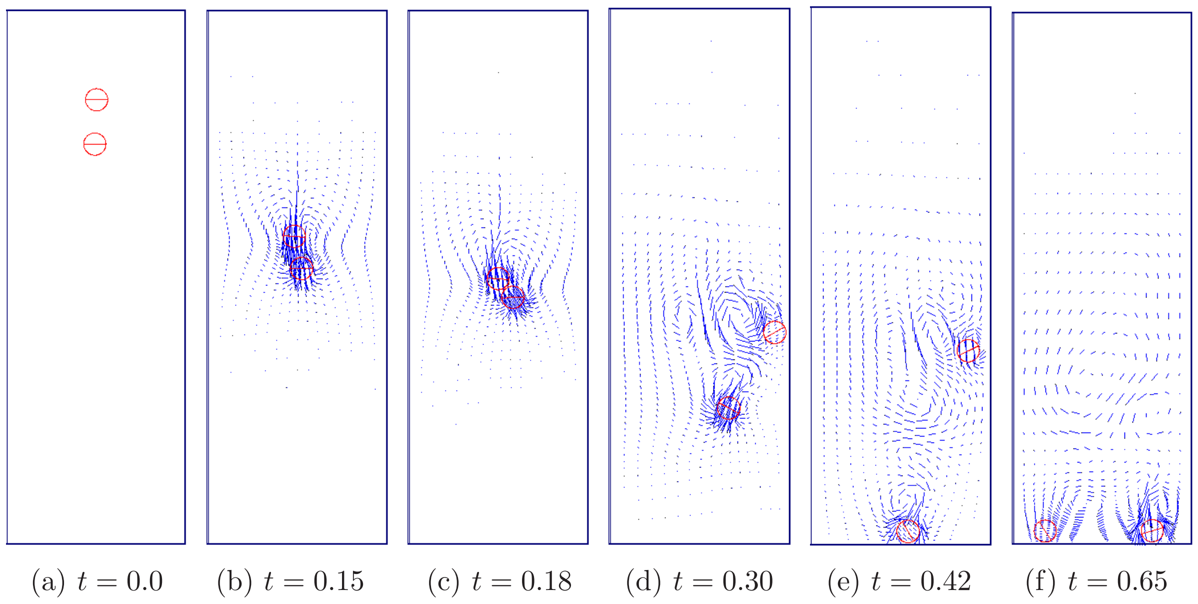
Figure 11: Induced rotation of a NACA0012 airfoil in a channel

When two particles are dropped close to each other, they interact by undergoing ”drafting, kissing and tumbling” [27], which is often chosen to examine the complete computational model of particulate flows, including the prevention of collisions. Therefore, we also study the sedimentation of two circular particles in a two-dimensional channel, comparing the results with respect



(a) $t = 0.0$ (b) $t = 0.15$ (c) $t = 0.18$ (d) $t = 0.30$ (e) $t = 0.42$ (f) $t = 0.65$

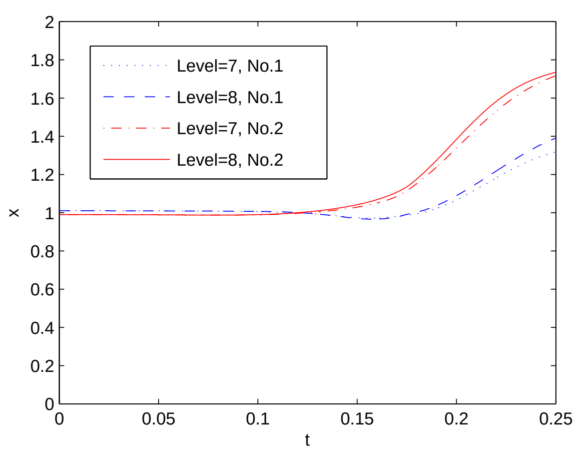
Figure 12: Deformation meshes for two circular disks falling down in a channel



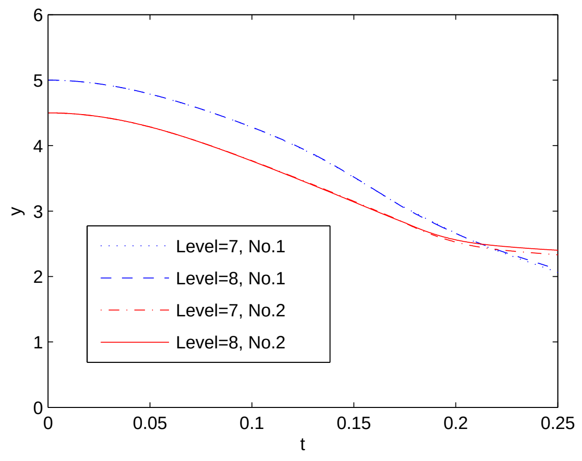
(a) $t = 0.0$ (b) $t = 0.15$ (c) $t = 0.18$ (d) $t = 0.30$ (e) $t = 0.42$ (f) $t = 0.65$

Figure 13: Vector fields for two circular disks falling down in a channel

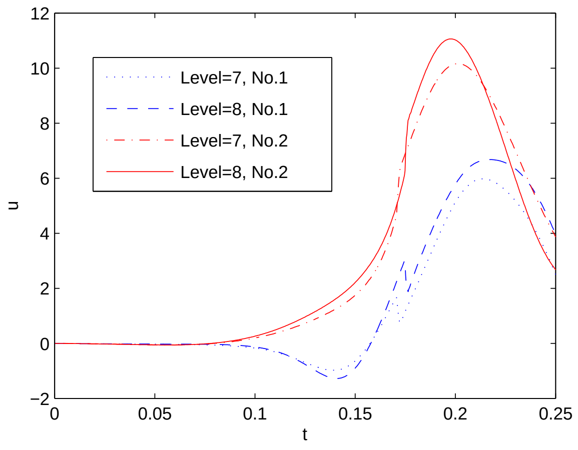
to two different level of grid deformation mesh sizes and regarding the results in [8]. The computational domain is a channel of width 2 and height 6. Two rigid circular disks with diameter $d = 0.25$ and density $\rho_p = 1.5$ are located at $(1, 5)$ (No.1 disk) and $(1, 4.5)$ (No.2 disk) at time $t = 0$, and they are falling down under gravity in an incompressible fluid with density $\rho_f = 1$ and viscosity $\nu = 0.01$, the gravitational acceleration velocity is $g = -980$. We suppose that the disks and the fluid are initially at rest. The simulation is carried out on the moving deformation



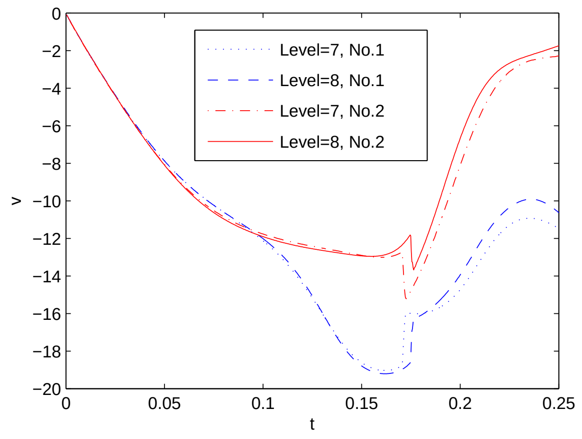
(a) x-coordinate



(b) y-coordinate



(a) u-component



(b) v-component

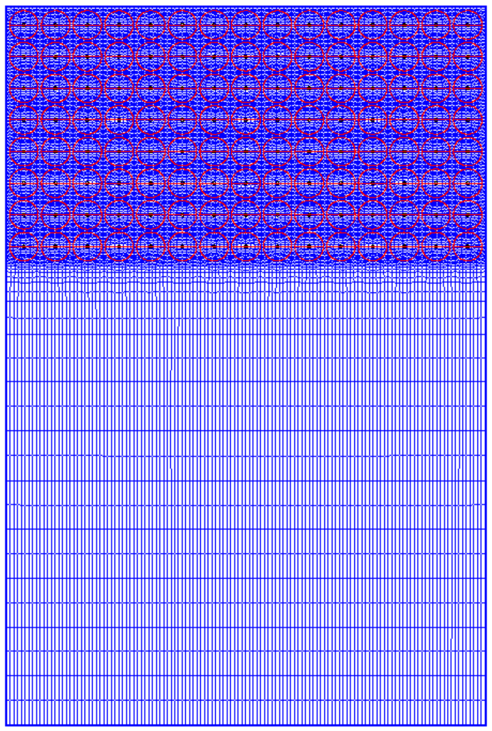
Figure 14: The time history of two circular disks falling down in a channel: (a) x-coordinate, (b) y-coordinate of the center of the two disks, and (c) u-component, (d) v-component of the translational velocity of the center of the two disks. Dotted line for No.1 disk and dot-dashed line for No.2 disk by deformation mesh Level $= 7$, as well as dashed line for No.1 disk, and solid line for No.2 disk by deformation mesh Level $= 8$

meshes, having two different level, i.e., Level $= 7$ with 49665 nodes and 49152 elements, as well as Level $= 8$ with 197633 nodes and 196608 elements.

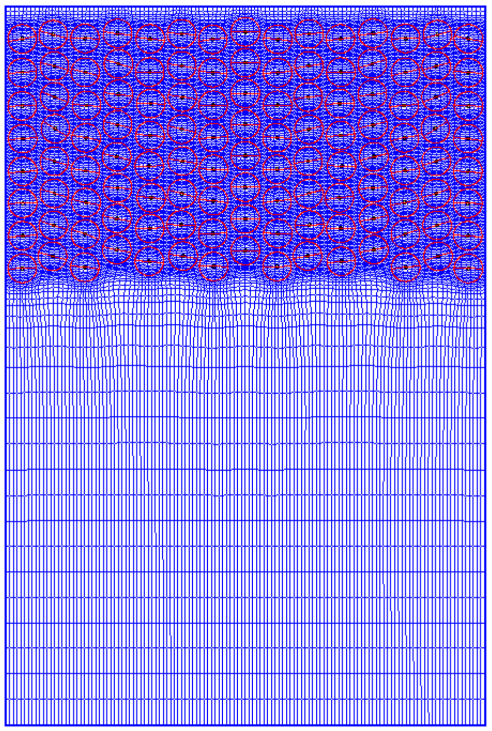
Fig. 12 shows the moving deformation meshes employed in the simulation of the two falling disks. The grid lines are always concentrated around the surfaces of the two disks and in the region of the gap between the two disks, and they move with the two disks during the computations. Fig. 13 presents the corresponding computed vector fields. From these figures, we can see that the disk in the wake (No.1 particle) falls more rapidly than the disk No.2 in front since the fluid forces acting on it are smaller. The gap between them decreases, and they almost touch (”kiss”) each other at time $t = 0.15$. After touching, the two disks fall together until they tumble ($t = 0.18$) and subsequently they separate from each other ($t = 0.30$). The tumbling of the disks takes place because the configuration, when both are parallel to the flow direction, is unstable. The No.1 disk is touching first the bottom wall at $t = 0.42$, while the No.2 disk reaches the bottom wall at $t = 0.65$. In Fig. 14, several quantities are plotted. These are the time histories of the x-coordinate and y-coordinate of the two disk centers, u-component and v-component of the disk translational velocities, obtained for the two deformation meshes, Level = 7 and Level = 8, respectively. We can see that the results computed on the two different deformation meshes are essentially indistinguishable. All results compare qualitatively well with those presented in [6, 8, 28, 29].

4.6 Sedimentation of 120 Circular Particles

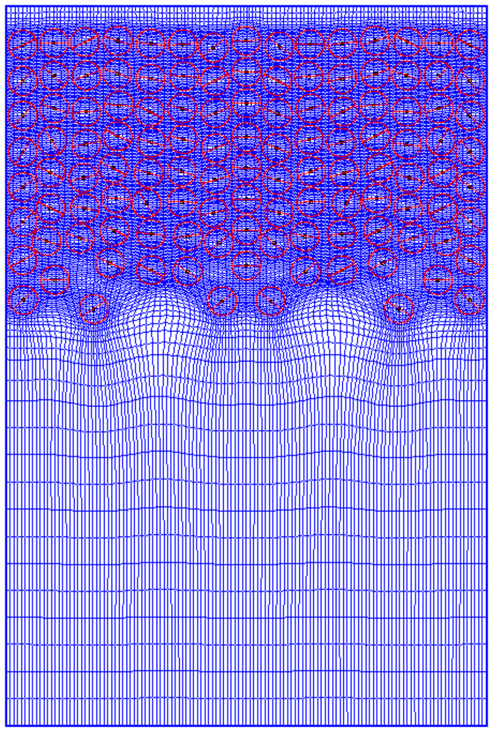
Finally, we consider the sedimentation of 120 circular particles with identical size falling down in a closed rectangular cavity. The width and height of the cavity are 4 and 6. The 120 particles are placed at the top of the cavity with 8 rows, while in each row the number of particles is 15. The diameter of the particles is 0.24. The range of the repulsive force is chosen as $\rho = 0.0167$. The position of the particles at time $t = 0$ is shown in Fig. 17 (a). The particles and the fluid are at rest at $t = 0$. The density of the fluid is $\rho_f = 1$ and the density of the particles is $\rho_i = 1.1$ ($i = 1, \dots, 120$). The viscosity of the fluid is $\nu = 10^{-2}$ (all quantities in non-dimensional form). The parameter ϵ_P in the collision model has been taken equal to 10^{-6}, and $\epsilon_W = \epsilon_P/2$, $\epsilon'_P = \epsilon_P$, $\epsilon'_W = \epsilon_W$. The moving deformation meshes are shown in Figs. 15 and 16. We can see that grid lines are moved and concentrated around the surfaces of each particle and in the regions of the gap between particles, and can always keep alignment with the surfaces of each particle even when the particle moves. The corresponding snapshots for the evolution of the vector fields of the 120 circular particle sedimentation are illustrated in Figs. 17 and 18. These figures clearly show the development of the Rayleigh-Taylor instability. This instability develops into a fingering and text-book phenomenon, and many symmetry breaking and other bifurcation phenomena including drafting, kissing and tumbling take place at various scales in space and time. Many complex vortices have been formed which pull the particles downward and mix each other. Finally, the particles settle at the bottom of the cavity and the fluid returns to rest.



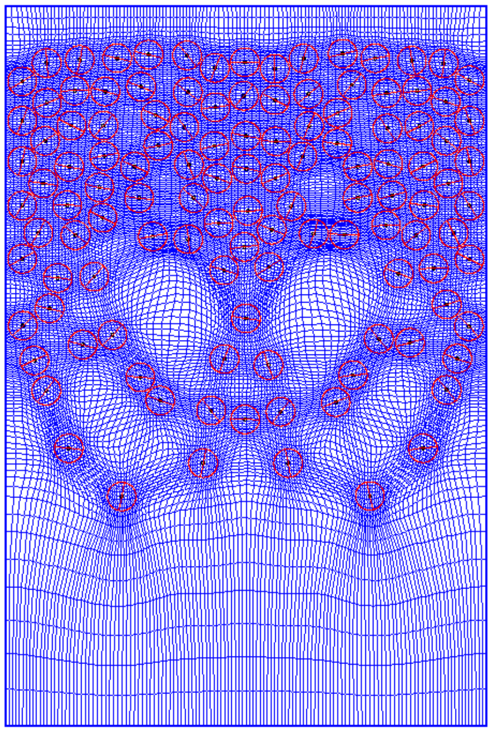
(a) $t = 0.0$



(b) $t = 0.35$

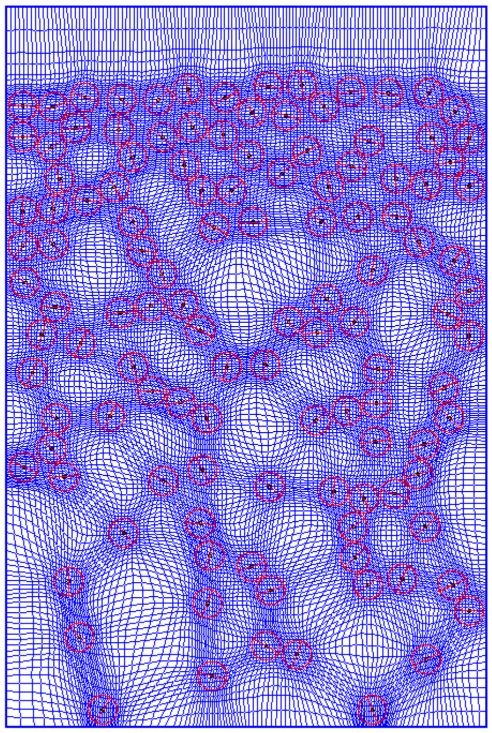


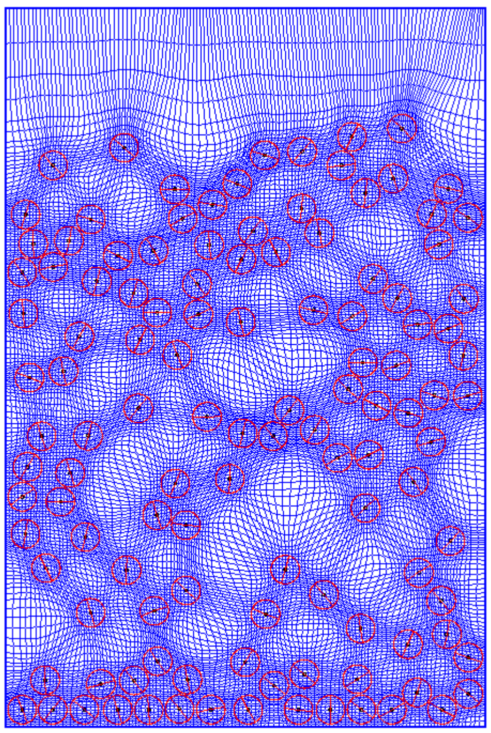
(c) $t = 0.56$

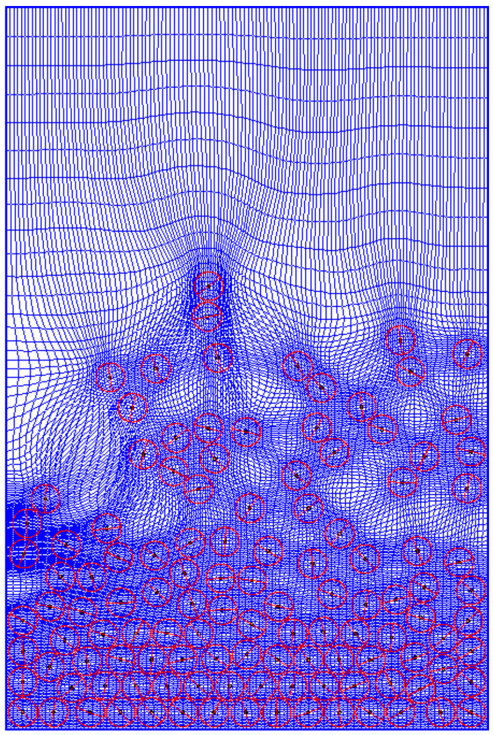


(d) $t = 1.02$

Figure 15: Deformation meshes for 120 particles falling down in a cavity


(e) $t = 1.53$


(f) $t = 2.04$


(g) $t = 3.06$

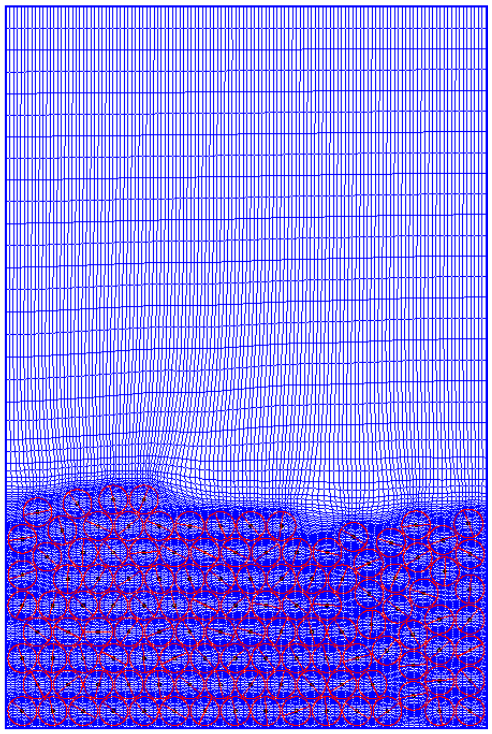
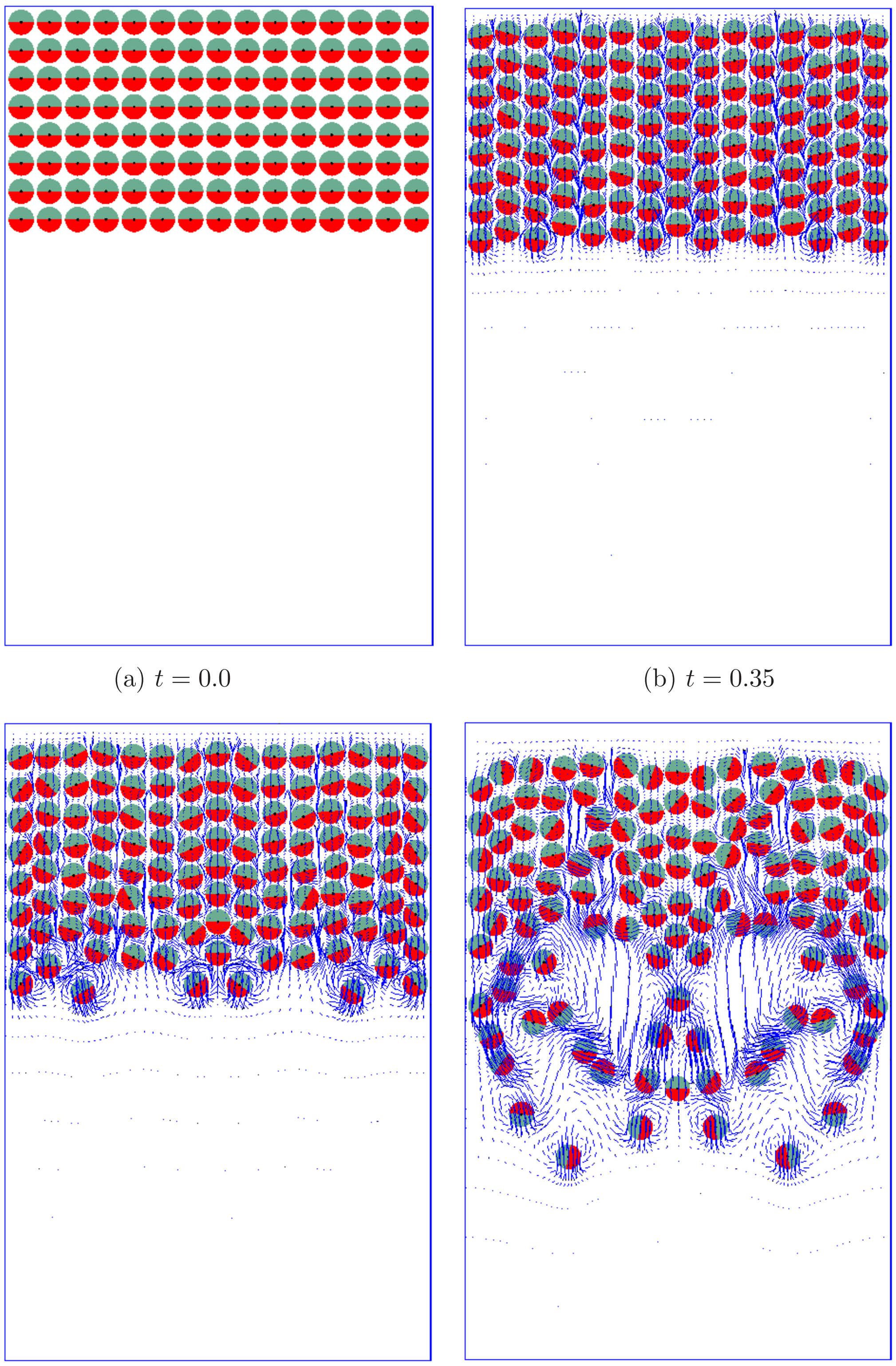

(h) $t = 5.0$

Figure 16: Deformation meshes for 120 particles falling down in a cavity (cont.)



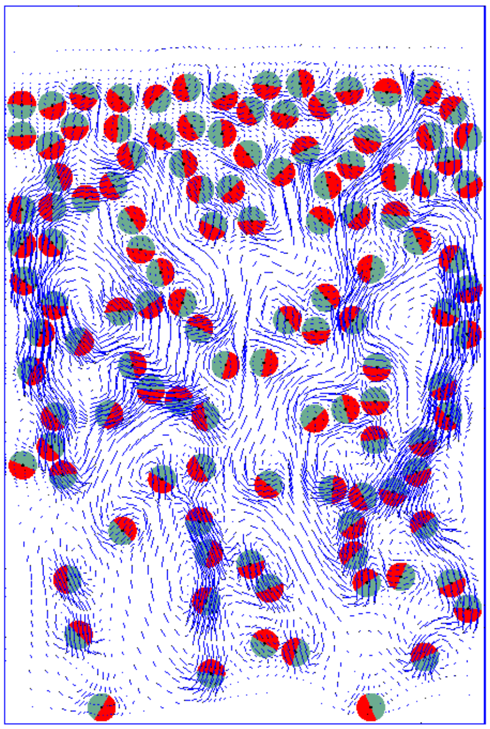
(a) $t = 0.0$

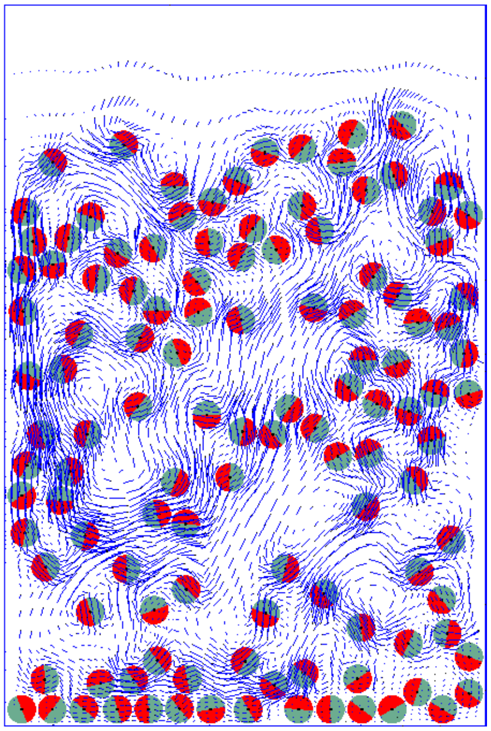
(b) $t = 0.35$

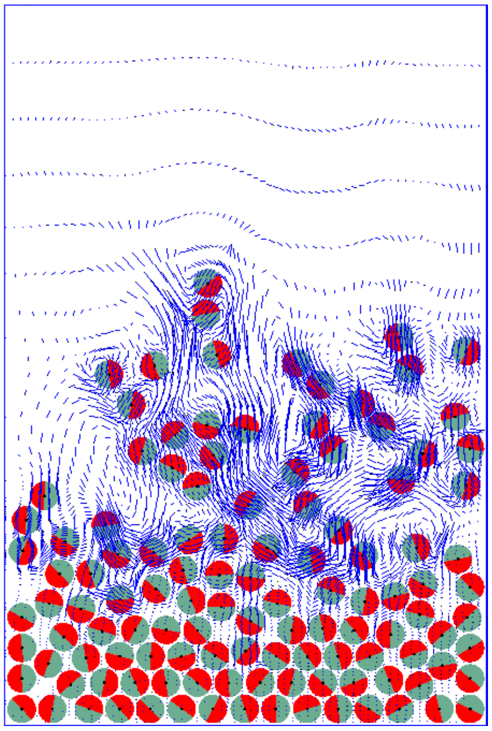
(c) $t = 0.56$

(d) $t = 1.02$

Figure 17: Vector fields for 120 particles falling down in a cavity


(e) $t = 1.53$


(f) $t = 2.04$


(g) $t = 3.06$

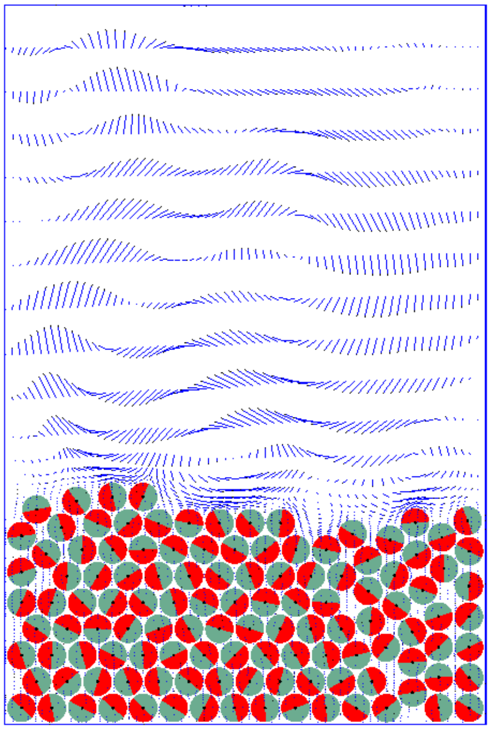

(h) $t = 5.0$

Figure 18: Vector fields for 120 particles falling down in a cavity (cont.)

5 Conclusions

We have presented the combination of the multigrid fictitious boundary method and the moving mesh method for the simulations of particulate flow with many moving rigid particles. The new approach directly improves accuracy upon the previous pure multigrid FBM for particulate flows. It is also computationally cheap and simple to implement. Since the size of computation and data structure of the moving deformation meshes are fixed, this enables the proposed method much easier to incorporate into most CFD codes without the need for the changing of system matrix structures and special interpolation procedures. It is suitable to accurately and efficiently perform the direct numerical simulation of particulate flows with large number of moving particles. One benchmark experiment, three numerical examples of single moving particle in a fluid as well as the drafting, kissing and tumbling of two disks in a channel and the sedimentation of 120 particles in a cavity have been presented to show that the presented method can significantly improve the accuracy for dealing with the interaction between the fluid and the particles, and be easily applied to real particulate flows with many moving particles.

References

[1] Hu, H.H., Joseph, D.D., Crochet, M.J.: Direct Simulation of Fluid Particle Motions. Theor. Comp. Fluid Dyn., **3**, 285 – 306 (1992)

[2] Hu, H.H., Patankar, N.A., Zhu, M.Y.: Direct Numerical Simulations of Fluid-Solid Systems Using the Arbitrary Lagrangian-Eulerian Techniques. J. Comput. Phys., **169**, 427 – 462 (2001)

[3] Galdi, G.P., Heuveline, V.: Lift and Sedimentation of Particles on the Flow of a Viscoelastic Liquid in a Channel. to be published

[4] Maury, B.: Direct Simulations of 2D Fluid-Particle Flows in Biperiodic Domains. J. Comput. Phy., **156**, 325 – 351 (1999)

[5] Glowinski, R., Pan, T.W., Hesla, T.I., Joseph, D.D.: A Distributed Lagrange Multiplier/Fictitious Domain Method for Particulate Flows. Int. J. Multiphase Flow, **25**, 755 – 794 (1999)

[6] Patankar, N.A., Singh, P., Joseph, D.D., Glowinski, R., Pan, T.W.: A New Formulation of the Distributed Lagrange Multiplier/Fictitious Domain Method for Particulate Flows. Int. J. Multiphase Flow, **26**, 1509 – 1524 (2000)

[7] Glowinski, R., Pan, T.W., Hesla, T.I., Joseph, D.D., Periaux, J.: A Fictitious Domain Approach to the Direct Numerical Simulation of Incompressible Viscous Flow Past Moving Rigid Bodies: Application to Particulate Flow. J. Comput. Phy., **169**, 363 – 426 (2001)

[8] Glowinski, R.: Finite Element Methods for Incompressible Viscous Flow. In *Handbook of numerical analysis*, **Vol. IX**, Ciarlet, P.G and Lions, J.L., Editors, North-Holland, Amsterdam, 701 – 769 (2003)

[9] Turek, S., Wan, D.C., Rivkind, L.S.: The Fictitious Boundary Method for the Implicit Treatment of Dirichlet Boundary Conditions with Applications to Incompressible Flow Simulations. Challenges in Scientific Computing, Lecture Notes in Computational Science and Engineering, Vol. **35**, Springer, 37 – 68 (2003)

[10] Wan, D.C., Turek, S.: Direct Numerical Simulation of Particulate Flow via Multigrid FEM Techniques and the Fictitious Boundary Method. Int. J. Numer. Method in Fluids, in press (2005). Currently, available on http://www3.interscience.wiley.com/cgi-bin/jissue/108061200 DOI: 10.1002/fld.1129

[11] Wan, D.C., Turek, S.: An Efficient Multigrid-FEM Method for the Simulation of Liquid-Solid Two Phase Flows. J. for Comput. and Appl. Math., in press (2005)

[12] Wan, D.C., Turek, S., Rivkind, L.S.: An Efficient Multigrid FEM Solution Technique for Incompressible Flow with Moving Rigid Bodies. Numerical Mathematics and Advanced Applications, ENUMATH 2003, Springer, 844 – 853 (2004)

[13] Berger, M.J., Collela, P.: Local Adaptive Mesh Refinement for Shock Hydrodynamics. J. Comput. Phy., **82**, 64 – 84 (1989)

[14] Huang, W.: Practical Aspects of Formulation and Solution of Moving Mesh Partial Differential Equations. J. Comput. Phy., **171**, 753 – 775 (2001)

[15] Ceniceros, H.D., Hou, T.Y.: An Efficient Dynamically Adaptive Mesh for Potentially Singular Solutions. J. Comput. Phy., **172**, 609 – 639 (2001)

[16] Bochev, P.B., Liao, G., de la Pena, G.C.: Analysis and Computation of Adaptive Moving Grids by Deformation. Numerical Methods for Partial Differential Equations, **12**, 489 (1996)

[17] Liao, G., Semper, B.: A Moving Grid Finite–Element Method Using Grid Deformation. Numerical Methods for Partial Differential Equations, **11**, 603 – 615 (1995)

[18] Liu, F., Ji, S., Liao, G.: An Adaptive Grid Method and its Application to Steady Euler Flow Calculations. SIAM Journal on Scientific Computing, **20**(3), 811 – 825 (1998)

[19] Cai, X.X., Fleitas, D., Jiang, B., Liao, G.: Adaptive Grid Generation Based on Least–Squares Finite–Element Method. Computers and Mathematics with Applications, **48**(7-8), 1077 – 1086 (2004)

[20] Grajewski, M., Köster, M., Kilian, S., Turek, S.: Numerical Analysis and Practical Aspects of a Robust and Efficient Grid Deformation Method in the Finite Element Context. submitted to SISC (2005)

[21] Dacorogna, B., Moser, J.: On a Partial Differential Equation Involving the Jacobian Determinant. Annales de le Institut Henri Poincare, **7**, 1 – 26 (1990)

[22] Turek, S.: Efficient Solvers for Incompressible Flow Problems. Springer Verlag, Berlin-Heidelberg-New York (1999)

[23] Turek, S.: **FEATFLOW**–Finite element software for the incompressible Navier–Stokes equations: User Manual, Release 1.2, University of Dortmund (1999)

[24] Turek, S.: A comparative study of time stepping techniques for the incompressible Navier–Stokes equations: From fully implicit nonlinear schemes to semi–implicit projection methods, Int. J. Numer. Meth. Fluids, 22, 987 – 1011 (1996)

[25] Turek, S.: On discrete projection methods for the incompressible Navier–Stokes equations: An algorithmical approach, Comput. Methods Appl. Mech. Engrg., 143, 271 – 288 (1997)

[26] Schäfer, M., Turek, S.: Benchmark computations of laminar flow around cylinder. in E.H. Hirschel (editor) Flow Simulation with High-Performance Computers II. Volume 52 of Notes on Numerical Fluid Mechanics, Vieweg, 547 – 566 (1996)

[27] Fortes, A., Joseph, D.D., Lundgren, T.: Nonlinear mechanics of fluidization of beds of spherical particles, J. Fluid Mech., **177**, 497 – 483 (1987)

[28] Singh, P., Hesla, T.I., Joseph, D.D.: Distributed Lagrange multiplier method for particulate flows with collisions. Int. J. Multiphase Flow, **29**, 495 – 509 (2003)

[29] Diaz-Goano, C., Minev, P., Nandakumar, K.: A Lagrange multiplier/fictitious domain approach to particulate flows, in: Margenov, W., Yalamov (Eds), Lecture Notes in Computer Science, Vol. 2179, Springer, 409 – 422 (2001)